23

The Abstraction, Transmission, and Reconstruction of Presence:
A Proposed Model for Computer Based Interactive Art

by

Christopher Nathan Dodge

B.A., New York University (1991)

Submitted to the Program in Media Arts and Sciences
School of Architecture and Planning
in partial fulfillment of the requirements for the degree of

Master of Science in Media Arts and Sciences

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 1997

Author.....
Program in Media Arts and Sciences
August 8, 1997

Certified by.....
Tod Machover
Associate Professor of Music and Media
Program in Media Arts and Sciences
Thesis Supervisor

Accepted by.....
Stephen A. Benton
Chair
Departmental Committee on Graduate Students
Program in Media Arts and Sciences

The Abstraction, Transmission, and Reconstruction of Presence:
A Proposed Model for Computer Based Interactive Art

by

Christopher Nathan Dodge

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning
on August 8th, 1997
in partial fulfillment of the requirements for the degree of
Master of Science in Media Arts and Sciences
at the Massachusetts Institute of Technology

Abstract

While there has been a recent growth of interest in the use of computers to assist in creative expression, there has not been much discussion about appropriate strategies for the design and implementation of such works. Many interactive computer works to date have used either cinematic or architectural approaches in order to provide for artistic content. It is my assertion, as outlined in this thesis, that there exists a new form of creative expression, via computers, outside of these other artistic formats - one that is purely "computational" in design and implementation. Through the following work, I will propose a general model for computer-based interactive art, using continuous "salient" parameters that are derived from the physical environment as sources for subsequent artistic transformation of abstract content. This work demonstrates several possibilities towards expressive cause-effect relationships between the interactive participant and the environment, illustrating these links through mathematical modeling and corresponding examples. Furthermore, this model of interactive art is upwardly scalable, allowing for multiple participants to interact within a shared environment. This thesis will explore some theoretical issues, present related research, introduce and detail each sub-system of the model, apply the model to six different real-world interactive environments as part of Tod Machover's *Brain Opera*, and provide an evaluation of the research project.

Thesis Advisor:

Tod Machover
Associate Professor of Music and Media

This research was sponsored by the *Things That Think* Consortium.

The Abstraction, Transmission, and Reconstruction of Presence:
A Proposed Model for Computer Based Interactive Art

by Christopher Nathan Dodge

The following people served as readers for this thesis:

Reader

Hiroshi Ishii
Associate Professor of Media Arts and Sciences
Program in Media Arts and Science

Reader

Bill Seaman
Associate Professor and Director
Imaging and Digital Arts Program, Department of Visual Arts
University of Maryland, Baltimore County

Acknowledgements

As it has been said, no man is an island. I have been blessed with the opportunity to spend time here at the M.I.T. Media Lab surrounded by so many inspiring and energetic people. I could never ask for a better set of collegues! My thanks - like the Media Lab itself - are wide and diverse:

Tod Machover, who graciously accepted to take me on as a Thesis Candidate. I salute your vision and courage to change the face of musical experience.

Hiroshi Ishii for volunteering to be a Thesis Reader and his general inspiration and poetry that is found in all of his research into Tangible and Ambient Media.

Bill Seaman for his friendship, guidance, and the many late-night discussions on interactive art. I admire your integrity and dedication in this field.

Andy Lippman for the fun RA position, for the unwavering support for the Scale Free Imaging project.

Sharon Daniel for her tireless work on *The Brain Opera* and for allowing me take part in the effort.

Joe Paradiso for being so tolerant of my un-announced office intrusions just to ask silly hardware questions.

The Brain Opera Crew for their dedication to do something that hadn't been done before.

To **Steve Waldman**, **Freedom Baird**, **Alex Westner**, **Rich Lachman**, **Joey Berzowski**, **Nitin Sawhney**, **Brenden Maher**, and **Karrie Karahalios**, for forming such a great group that has become a legend – the **Thursday's Night Gang**. One could not ask for a more fun set of buddies with which to sample Boston's cuisine and bar offerings.

My officemates **Pascal** and **John (Wad) Watlington** for moving the couch out of the office so that I could burrow there for a little time.

To **Bill (Bubu) Butera** for all of his encouragement and advice.

To everyone else in **The Garden** for making such a great work environment and rightfully making The Simpsons a high priority in life.

To everyone in the **Opera of the Future Group** who took me in as a stranded orphan and treated me as one of their own.

To everyone on the **Shadow Masks Softball Team** for proving that even being the worst can still mean having lots of fun.

The Kendall Cafe, the ***best*** hole-in-the-wall I've ever frequented, for all the great food and beer selection that helped be retain my sanity during the stressful times. Keep the Sierra Nevada flowing, amen!

To all of the **Things That Think**, **Television of Tomorrow** and **Digital Life Consortia Sponsors** for their financial and equipment support. Also thanks for their interest in my research work and for the feedback during demos.

.....and lastly, my utmost special gratitude goes to my soon-to-be-wife **Birgit Locher** for all of her love and tolerance shown to me during my surly moods throughout this Herculean effort.

Contents

List of Figures

List of Tables

Chapter

1 Introduction

" The medium is the message."
- Marshal McLuhan

" The message is the message."
- Walter Bender, M.I.T. Media Lab

Consider the following scenario: you are walking through a train station on your way to work in the morning. Looking over the throngs of fellow commuters, you notice a line of video projections along the walls of the building. Mixing in with the crowd is a light, airy video projection of colorful, floating streams that hover above the crowd, moving hurriedly alongside of the hectic mass in the same direction as everyone else. Your eyes catch a long gaze at the kinetic video sculpture situated in this public space. Running a little bit behind schedule, you forge ahead along with the rest of your compatriots.

Evening arrives and you return to the same train station to leave the city. Hoping to be able to spend a little more time looking at this public art work, you return to the same location as in the morning. However, everything is different in the work, the video sculpture that was earlier moving right to left, is now moving in the other direction and the formations of the bands of color are now slower, stretching out across the screens. You are a little puzzled, thinking about what has changed with the video work, as it obviously must be a simple video projection. Although the content of the work is different, it still is eerily and appropriately fitting to the environment which now has a trickle of fellow evening commuters who are moving left-to-right behind you. After pausing a little while, you deduce that the video maker was smart enough to time his/her work to reflect the changes of the environment from morning to evening. What a long videotape it must be to play out this daily rhythm in a linear manner!

As the next day is a weekend, you decide to come into Boston to take in the sights. This time when you arrive in South Station, the video work is again very different than the night before. The flowing bands of color are very abstract now, lacking a distinct formation of motion, swirling around themselves. Again, although the work has changed, it is nevertheless fitting to the public environment where a few tourists are meandering around the building. This can't be so, you ask yourself, it's just not possible to make a linear video work correspond to the immediate environment like this! It seems as though the art work adapts itself to the activities within the real physical space. What is going on?

Actually, the above is the scenario of a recently produced interactive installation named *What Will Remain of These?* (1997) which has been shown at a number of exhibitions throughout the world. In this work, a set of hidden video cameras "deconstruct" the motions of a large number of people in a public space into a series of numerical qualities. These qualities - such as motion direction, velocity, and color content - are then projected back into the physical environment through a physical based particle simulator. Whatever motion patterns the unwitting participants make, through their daily rush-hour behaviors, these are mimicked by the system. The work is intended to reveal large-scale patterns that we are all a part of as individuals. The process of rhythmic motion is adapted by the system, playing it back to us in a highly abstracted form before our eyes. As there is no clearly delineated space

where one explicitly interacts with the work as an individual, the piece strives to give a more mysterious effect through the use of implicit and group interactions. The goal is to create a mirror image of ourselves, albeit a hazy and distorted one.[1]

This work is an example of what is known as an interactive computer art installation. While the notion of interactive computer art has been a subject of great experimentation and theoretical discourse[2], it has been very difficult to design and implement such works in a formal and mathematical fashion. This is due to the unfortunate division between the artists and technologists, each having their own form of discourse.[3] Furthermore, many interactive works tend to use a cinematic language - i.e. actors, scenes, cameras, etc. - in order to communicate a thematic idea. Virtual Reality, and other similar formats, create new digital worlds and allow the viewer to objectively navigate through the space.

However, this above described work uses a different model of interactivity in which there is no navigation of a constructed digital environment, as there is no explicit content that has been programmed. The digital environment inherits the qualities, i.e. bodily motion, of the physical environment, rather than the viewer learning the rules of the interactive work. We could say that the interactive environment mathematically maps the continuous and qualitative descriptions of the physical world into a computational one, which in the above example, re-visualizes the unfolding process of mass transit over time. Such a work of interactive art is meant to serve as a example of this thesis' model of interaction. In such an interactive environment, the computer creates a set of continuous-valued sensory descriptions of the environment, performs a series of well-defined mathematical transformations of these descriptions, and remaps these salient descriptions to an output system.

All in all, I hope that such a model of interactivity will be able to provide another counterpoint to the McLuhan argument: *"The mapping is the message."*

1.1 Motivation and Thesis Contributions

My interest in this thesis document, and set of described projects, is to provide a formal and well-defined set of bridges between computational technologies and artistic expression. As there are as many approaches to art as there are artists, my intent is to explore that which I term a *model of computational art*. Computational art differs from what I would call *computer-mediated art* for one fundamental reason: pure computer art could not exist in other technologies. Computational art is algorithmic, using models and simulations to reveal the thematic content that is encoded into the system, exploiting all of the capabilities that computers offer. This system is meant to contrast with traditional multimedia uses of computers that focus on the communication of a cinematic, narrative experience. While both methods use sound and image technologies, cinematic experiences could be found in other formats and therefore cannot be considered pure computational art, in my opinion.

Computer mediated narratives, as illustrated by Brenda Laurel in [4], tend to be figurative in design, using complex objects such as characters, setting, and plot to literally communicate a theme. In my proposal for a computational art, I prefer to portray a process that would communicate *qualities* of an experience. In this *exposition of process*, the viewer would be led to a particular reasoning rather than being explicitly told the thematic concepts. For example, rather than creating a cinematic environment, through a series of time-elapsed sequences, that

illustrates traffic flow in a city, we could create an algorithmic model that creates a visual flow within the digital environment, using a simulation of real-world observations. The cinematic experience is fixed and static while the algorithmic model, albeit more abstract and open to interpretation, can dynamically respond to real-time data. Both methods communicate the same phenomenon, but the latter, in my opinion, is more consistent with the use of abstraction in 20th century art. Therefore, in the projects and theoretical model described in this thesis, I wish to capture and reproduce what I term *salient characteristics* of both the thematic meaning as well as the interactive viewer.

In order to achieve these goals, I have created a general model, that attempts to break the design and implementation of interacive systems into a series of formal and mathematical approaches. The model has three components, described in Chapters 4, 5, and 6, that use a set of continuous valued parameters, named *salient vectors*, to drive the interactive system. These salient vectors are initially computationally derived - or "abstracted" - from sensing technologies that are "viewing" the interactive participant. Several possible abstractions are proposed and implemented in this thesis document, such as color, form, motion, etc. This user representation is considered a point in a hyper-dimensional interface space, subject to subsequent mathematical transformations and other operations as the vector proceeds down the system pipeline. Eventually this salient vector is used to control an output system that maps this vector into a "reconstruction space", creating the output that is feed back into the interactive environment. This thesis document presents several key reasons, as exemplified by the described projects, for the use of mathematical transformations as a method for building interactive systems.

The fundamental issue in this thesis, both in terms of artistry as well as technology, is what I call *presence*. This keyword, used centrally in the title of this thesis, is somewhat difficult to define. This is where the artistic interpretation comes in: How do we communicate the existence of a thematic idea within such an abstract notion of a computational model? How do we represent the physical presence of an interactive participant? What presence does he/she have within the interactive cause-effect mapping? If the computational environment is multi-user, how is one person aware of the presence of another? Finally, what is the relationship between all of these entities and the output of the system?

These are several complicated questions that need to be answered. Unfortunately, there is no unified formal research approach available at this point in time. However, it is possible to make suppositions and experiments based on these notions in order to get a better idea of how to approach the design and implementation of computational art. Through this thesis document, I will propose a general system which tries to capture and communicate salient information through a parameter-based system that creates cause-effect mappings between the user and the interactive environment through mathematical transformations. In order to be comprehensive, it is important to compare and contrast these techniques with other methodologies current in place within the field of interactive art.

The remainder of Chapter 1 will be devoted to a cursory overview of interactive systems. Chapter 2 familiarizes the reader with several related research and development projects, both previous and concurrent. Chapter 3 presents a brief introduction to the proposed interactive model. Chapters 4, 5 and 6 discuss, in detail, the sub-components of the proposed model of interactivity including input sensor, data communication, and output visualization

technologies and methods. Chapter 7 presents the implementation of these models into actual working systems which are part of Tod Machover's *Brain Opera*. Chapter 8 extends the *Brain Opera* implementation, improving several weakness in the original design. Chapter 9 evaluates the subjective success of the project, proposes future research and development directions, and serves as a conclusion to the thesis document.

1.2 Interactive System Theory

The following sections present a summary of the main discourses surrounding contemporary interactive media art, drawing from the current literature and discussion foci that have dominated the field for the past few years. As the theoretical study of interactive art is relatively new, many of the following topics are still evolving and subject to interpretation. To me, the most interesting aspects of interactive computer art lies in the experimental combination of both dynamic system design and artistic themes that can be effectively delivered within such a structure.

This leads to one of the fundamental questions that audiences have towards interactive art: "*Why is interactive art interactive and "traditional" art non-interactive? Every time I look at the same "traditional" piece I am interacting with the vision of its creator and the experience varies from viewing to viewing depending on my mood, its placement alongside other works at the exhibition, and how it is described in the program notes, etc. Therefore it's meaning or significance is indeed non-linear and highly evolving.*"

What is unique to an interactive artwork is the systematization - usually through computers - of a set of rules and relationships between the viewer and the content of the work. It is through interactivity that we can change the artistic "surface" that is presented to the audience. Different surfaces - so the hope - will produce qualitatively different experiences in the impressions of the audience. It is important to note that "meaning" and "significance" are highly subjective experiences that the viewer receives when viewing any work and therefore very difficult to predict and formalize. In a way the interactive artwork is incomplete without the interactive participant, fulfilling Duchamp's statement "the viewer completes the picture." [5]

1.2.1 The Creator's Creator

As noted in William Mitchell's The Reconfigured Eye, the ultimate aesthetic of computer-based art stems from the computational tools that are employed by the artist.[6] Unfortunately, the knowledge of computer systems programming is not widely distributed, forcing people into either software user or developer roles. When the artist, who is not him/herself a computer programmer, chooses to develop an artwork with a set of computational hardware and software tools, there is a corresponding set of aesthetics that are inherited with this choice. There is no such thing as an aesthetically neutral set of software products, as the designer of the tools has already embedded his/her particular vision of what "computer art" should be. A good example is with Virtual Reality technologies, which have a rather large economic force supporting the research and development of basic communication technologies. This can be an asset to the artistic community, as such a large market base gives the technology the possibility of becoming a high volume/low margin business. Indeed, with the advent of several low-cost 3D graphics accelerator

peripherals for the IBM PC, the power that was once reserved for the most expensive systems is now within the reach of many low-budget producers.

However, this is a double-edged sword. These VR technologies are based on a simple perspective model that is intended for the replication of wire-mesh polygon structures with simple shading algorithms or texture mapping. Aside from these features, these systems cannot produce anything else. This is a clear indication of the limitations of using commercially available products, as the content developer is aesthetically constrained to that which is considered economically viable. If the artist employs such a software tool set, it should be out of free choice with the willingness to work within that aesthetical format. However, as interactive environments rely on new and novel devices for the presentation of audio-visual material, there is the danger that certain styles become rapidly cliched and loose much of their artistic appeal. This sentiment is echoed by Graham Weinbren[7] and Tamas Waliczky[8], who call for specialized systems for artistic environments that are outside of economic forces.

1.2.2 State-space Content Models

Interactive art is largely the systematization of the representations of the viewer, the underlying content, and the relationships between these two elements. Here, I will present two contrasting formats of content representations. The first model type uses a finite-state space to represent a set of state potentials and transition events between the states. Such a model is widely associated with non-linear narrative systems and has a relatively simple representative structure. In Figure 1-1, there is an example finite-state space where the circles represent states of the system and the arrows indicate the paths of transitions that are allowed between states. In a completely connected state-diagram that has N number of unique states, where each state can transition to all other states, including itself, there are N^2 possible transitions. Therefore, as we increase the number of allowed states, the number of transition events grows exponentially. Of course, a completely connected state graph is not always needed and we can generalize the number of transitions to T*N where T is the number of transitions from a state and N is the number of states, and, obviously, $T \leq N$. However, there still are several production problems when the number of states and transitions becomes very high.

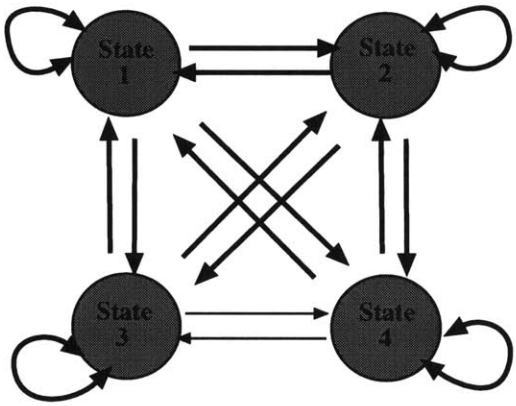

Figure 1-1. Simple state diagram with 4 states that are fully interconnected.

The canonical examples of a state-space mode of interactivity are the "choose your own adventure" books that were briefly popular in the early 1980's. Here the reader would read the contents of a page or two, usually with an adventure or fantasy theme, and then be presented a list of options that his/her character could make, typically limited to two or three at a time. There would be a page number listed that the reader should jump to, should he/she choose to take that course of action, making the reading of this book non-linear and interactive in a very rudimentary sense. The states in this environment are pages that relate to a spatio-temporal placement of the protagonist of the story and the transitions are made between pages in the book that keep the plot moving forward.

This genre of storytelling is very important in terms of state-space models of interactivity for several reasons. First, the story is arranged such that the reader receives a perceived continuity of the narrative, although the data (the text) is non-sequential in the database (the story). Second, the reader can only make decisions at certain points in the story, thus the interactivity is only during select times that denote windows of opportunities to alter the path of the narrative. Therefore the interactive experience is not continuous in nature. Third, the number of choices is constrained by the size of the storage medium (the book), due to limiting factors such as printing expense and the costs to pay writers to develop so many narrative threads. Last, there is only a binary representation of the reader in this system, as we must choose one out of two or three courses of action. It is not possible to veer away from these state transitions.

Needless to say, this genre of interactive art did not outlast its novelty after a few years. CD-ROM projects that have adopted a similar format also often run into such problems, creating calls for a model-based approach[9]. However, with the introduction of the World Wide Web as both a communication medium as well as a viable programming environment, a new state-space genre is currently being explored by several researchers and artists. As the Web is a highly distributive bi-directional environment, several new works, such as the *World Wide Movie Map*[10] and *Dream Machine*[11], allow the state-space to be developed as well as navigated by the audience. This format may prove to be an interesting avenue to follow and hopefully additional experimentation will be done in this area.

1.2.3 Parametric systems

In contrast to the state-space representation of interactive experiences, a parametric system is not concerned with permutations. By parameters, I am referring to run-time variables for computational models that alter the output of the system in a seemingly continuous manner. As one might expect with such a term, parametric systems are largely mathematical systems rather than higher-level data representations such as state diagrams. For example, take the mathematical expression:

$$y = f(x) = mx + b \tag{1-1}$$

where m and b are arbitrary constants. Here, in this function, the independent variable x is related to the output of the system based on these two parameters, m and b.

What then does x *mean* in this system? This raises the fundamental question of parametric systems as there are no inherent semantics that bind a meaning to their representation. All that parametric systems do is numerically map values from one symbolic space to another, from a 1-space to another 1-space, under the control of these two

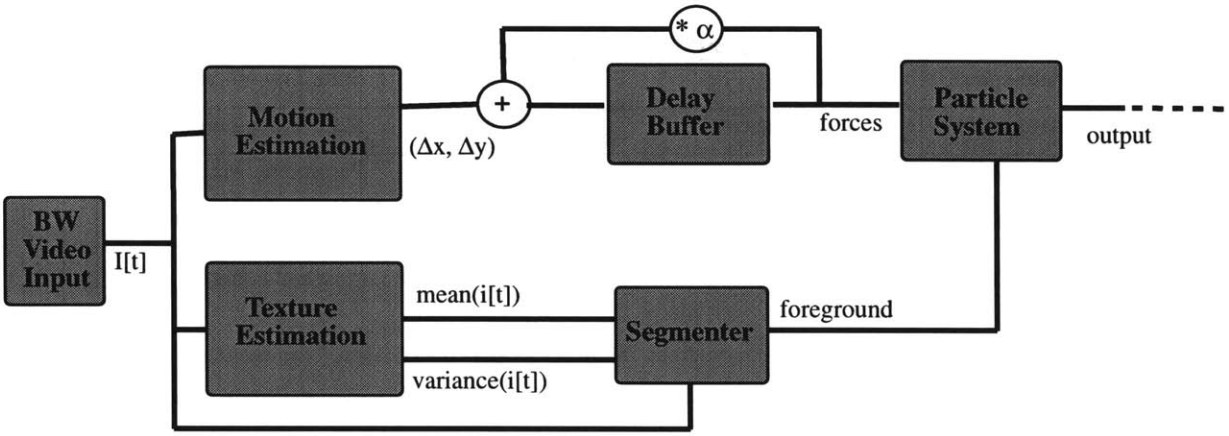

Figure 1-2. Sample input/output links in a parametric system.

parameters, as in the above example. Any impression of meaning is ultimately created by successive mappings that reach the subjective viewer and will be discussed in later chapters. What makes such a method interesting is that the system is succinct and well-defined, but is able to produce significantly different outputs based on the values of the parameters of this equation.

Another advantage of using a parametric system is that the output of one component of the system can be used as control parameters to another - perhaps higher-level - section of the interactive environment. Consider a system block diagram as depicted in Figure 1-2 which demonstrates how the output from one system is used as input into another system. Such a method of input/output connectivity is similar to the way Opcode's MAX software program allows the user to form MIDI and sound environments. These parameters, as presented in this thesis, are derived from a series of interfaces abstractions that map the viewer into the environment. The interactions of the viewer become the control parameters into the entire system, creating a cause-and-effect linkage between the stimulus (the actions of the viewer) to the response (the change of the system). Therefore this type of system is continuously "reacting" to the user, unlike state-spaces that limit the user interactions to discreet transitions between states.

1.2.4 User Modeling and Representation

Given that the viewer is one of the central components of an interactive environment, we need to consider the corresponding representations of the participant. First we should examine what the interactive environment contains in terms of thematic content and physical incarnation. Is the viewer alone or in a group? Is he/she among tangible objects with which he/she will have intellectual or emotional associations? To what extent can the viewer alter the presentation of the work? How can we make for a consistency between the viewer and the virtual world, in which the actions of the audience ultimately map to an appropriate response of the system? These are important questions because they create links between the members of the audience and the interactive environment. For example, if a viewer is placed in an empty room with only video projections on the walls, there are no points of interactive reference that give indications of the role or purpose of the interactions. Otherwise, if the user is given a wide set of

Content Representation	User Representation	Transition Source	Cause-Effect Mappings
State-space	Binary	User Events	State Transitions
State-space	Parametric	Classification	State Transitions
Parametric	Binary	Look-up tables	Parametric Mappings
Parametric	Parametric	Continuous Input	Space Transformations

Table 1-1. Four possible representations of the interactive viewer and content.

physical objects that act as interface devices, such as Jill Scott's opened suitcases,[12] the range of associations of the viewer are directly related to the objects themselves. In the former case, the user interaction is modeled in a context-free environment, whereas in the latter he/she navigates the meanings of the interface objects themselves.

Much like the sub-section on content representation, the user can also be modeled either in a binary state format or through derived continuous parameters. This gives us four possibilities of system integration which are outlined in Table 1-1.

A binary representation of a user is typically a multidimensional binary vector. Each element of the vector is associated with a particular state of the user. For instance, let us take the example of four null-dimensional input devices, such as switches, one of which can be pressed by the viewer at any point in time. This would give us a representation of the user as such:

$$\mathbf{u} = (s_1, s_2, s_3, s_4) \tag{1-2}$$

where s_1, s_2, s_3, and s_4 are the binary state of each of the switches, either a 1 (on) or 0 (off). This binary representation has a fundamental problem because of its non-linearity, as it is not possible to perform simple mathematics on such a format. For example, consider two user states of four switches $\mathbf{u}_1$ = (0, 0, 1, 0) and $\mathbf{u}_2$ = (0, 1, 0, 0). It is not possible to compute the mathematical average of these two representations, i.e. $0.5*(\mathbf{u}_1 + \mathbf{u}_2) \neq (0,0.5,0.5,0)$. The invalidity of such simple mathematical operations creates problems when applying such representations to an interactive system, as will be demonstrated later in Chapter 5.

As a contrast, Figure 1-3 shows a parametric description of a viewer, based on Jim Davis' work on gesture recognition through machine vision. Here the user is modeled as a set of orientations of three main components of his/her body, his/her torso, upper, and lower legs. Such a representation could be:

$$\mathbf{u} = (\theta_1, \theta_2, \theta_3) \tag{1-3}$$

where θ_1, θ_2, θ_3 are the relative joint angles between these body components in radians. Although the binary representation is four-dimensional, the parametric description, due to its continuous values, allows for a much wider set of variances at the input. The binary representation only provides for four (or 16 if we allow multiple switches to be used at once) unique values that describe the user. However, this parametric representation can theoretically contain an infinite number of values, limited by only the computer data types – e.g. single-precision versus double-precision floating point values - provided by the computer. Furthermore, as these vectors are both continuous and orthogonal, they could be considered as mathematical points in an abstract "space", as will be described in Chapter 3.

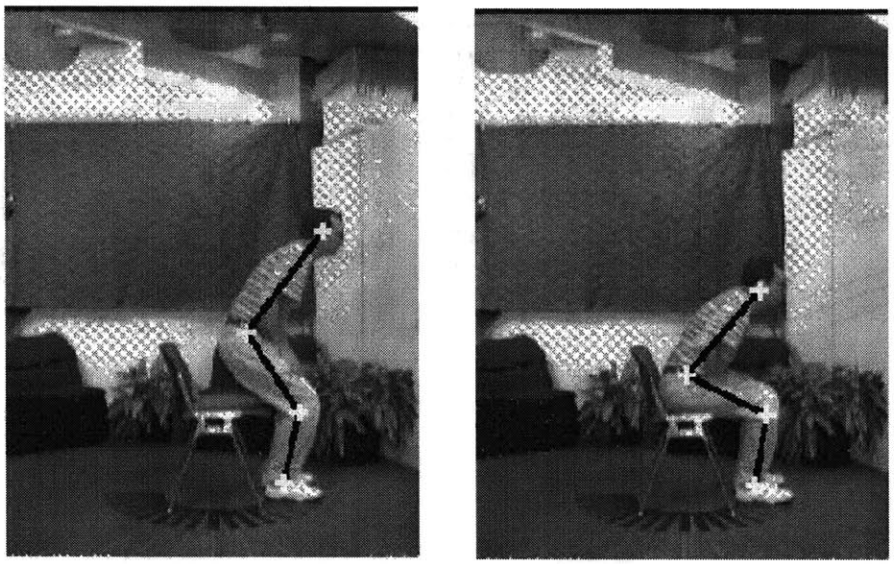

Figure 1-3. Three dimensional joint-angle parametric representation of user. Images taken from [13].

More importantly, we can perform mathematical operations such as addition, subtraction, multiplication, and division in order to create systems that can support a computational form of artwork. Since this thesis uses mathematical transformations to create input/output relationships, it is important that the content as well as the user are represented by continuous values. As, in this system, inputs and outputs can be linked arbitrarily together, we need to be able to guarantee, over at least a certain range, the validity of mathematical operations. This is to say that given two user representations, $\mathbf{u}_1$ = (0.2,0.6,1.0) and $\mathbf{u}_2$ = (0.8,0.2,0.5), the average representation can be mathematically defined as $\mathbf{u}_a = 0.5*(\mathbf{u}_1+\mathbf{u}_2)$ = (0.5,0.4,0.75). What such an average "means" will be formally explored in later chapters.

1.2.5 Constructivism in Interactive Art

"Modernism is dominant but dead."[14]
- Jurgen Habermas

A symposium on new media – including interactive art - took place at the MultiMediale 4 (April 1995) at the Zentrum für Kunst und Medientechnologie (ZKM) in Karlsruhe, Germany. The art theory symposium was named the "Second Modernism". In the symposium talks, several prominent art theorists presented ideas on how computer technologies, through interactive media, are heralding a resurgence of the ideals of the Modernist era of the early 20th century.[15] The Modernist era, although difficult to succinctly define, is typically associated with Enlightenment movement and the Industrial age where the individual is believed to have objective understanding of the environment around him/herself. The individual can understand the world through his/her senses and re-create the world through physical systems that model reality. Therefore, the title for the symposium is fitting, as these modernist attitudes are typical for contemporary interactive media, in which the artist creates artificial worlds that are navigable in the same sense that one walks through architectural spaces. Physical space is directly translated into digital space.

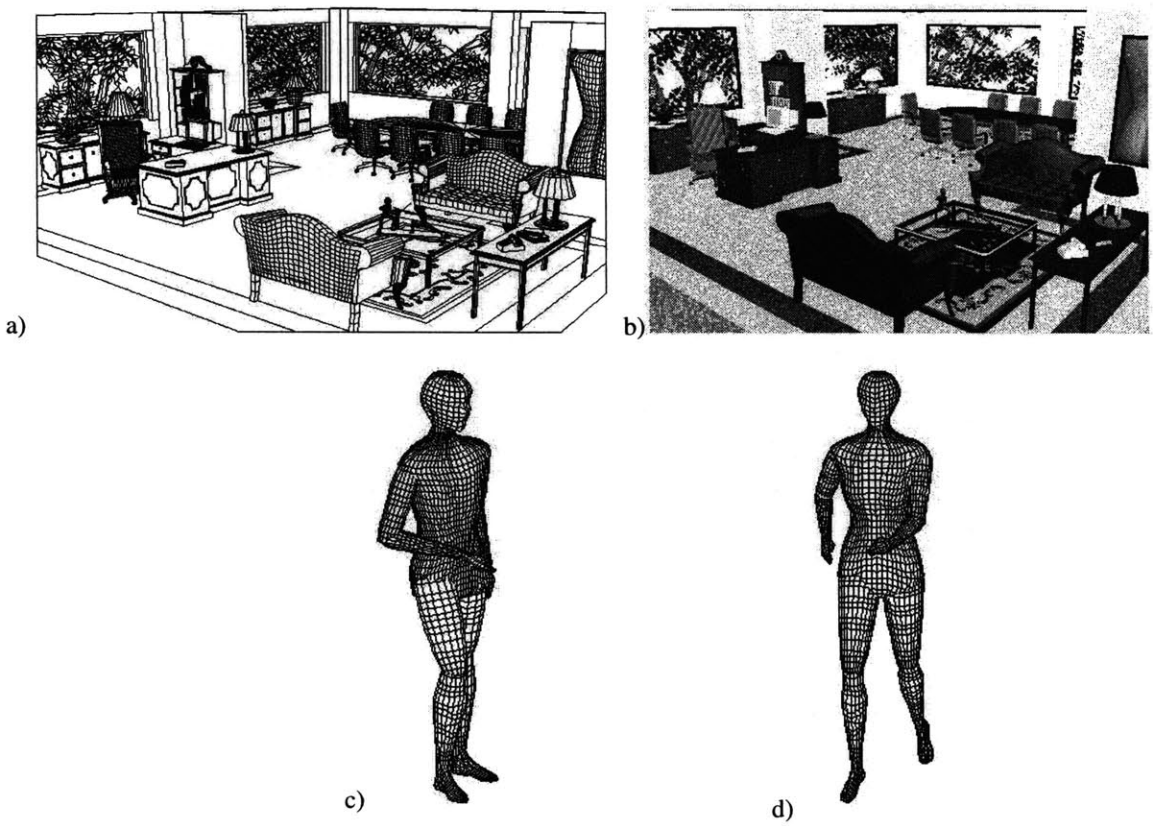

Figure 1-4. Examples of polygon meshes (a, c and d) and one (b) rendering that attempt to "realistically" portray real-world objects. Images taken from [16].

Constructivism, borrowing the term from Mitch Resnick in [17], is a process in which the computer artist uses a set of media "primitives" that can be combined in order to make ever more complex structures. Creation is a "bottom-up" endeavor, beginning with a blank canvas, gradually making additions to the environment, grouping together several primitives into objects, and collecting objects in order to make a scene. Each layer of the object hierarchy is comprised of a group of lower-level objects, organized in such a way that the collection of primitives operate in conjunction with one another. The best example of such a system is computer graphics where the fundamental atom is the polygon. Figure 1-4 shows how polygons can be grouped together to form object meshes that can be connected to form articulated objects, and so on.

So out of "digital nothing" comes a constructed world, hence the name constructivism. Whatever is found within the interactive environment is the product of the artist/engineer, leaving very little to chance. While such a process does indeed sound promising, it is very difficult to create artificial worlds that both intrigue and delight. Much like the discussion about state-space systems, constructivism requires that any richness or variance within the interactive experience is explicitly pre-programmed into the computer. As Frances Dyson writes, one can only experience that which has been constructed in the virtual, therefore there is an inherent limitation in the range of possibilities:

> "This movement towards objectification, delineation, and fragmentation subverts the 'liberatory' space virtuality seems to offer. ... Yet the parameters of this 'space' are ultimately defined by the binary yes/no, on/off commands of digital systems, designed to receive and process information according to set codes that are themselves not neutral. Rather than entering a 'free-space,' subjectivity is recontextualized within the programmatic grid of technology, and embedded in this grid are all those elements that drive the fix and rigid reality, the prescribed subjectivity one might, through VR, be trying to escape."[18]

The fundamental problem I see with constructivism is the attempt to create an artificial world which entails all of the richness that we expect from artistic environments. In computer graphics, there as been a tendency to create realistic images that accurately model the real world, expressed in the term "photorealistic". Taking this coined term apart, it would seem as if the goal of computer graphics, in general, where to make the computer generated image as "real" as a "photo" of the same scene. The irony, of course, is that there already is a sensory rich environment that has been developed: our own physical environment around us. With constructivism there is the overarching attempt to recast the physical world, down to the minute details, into the digital machine. It is my intent, in this thesis document, to suggest mechanisms that can use the dense sensory information in the real world, through the abstraction component which will be described in Chapter 4, to drive an artificial world. The goal is to derive "top-down" qualities that would be difficult to build "bottom-up".

One notable exception to this brief discussion is Tamás Waliczky's delightful computer animation, *The Garden*. In this work, Tamás Waliczky developed a new perspective system that subverts the entrenchment of Renaissance perspective systems found in commercial 3D animation packages. The research and development was, however, motivated with an artistic theme in mind, that being the egocentric state of a young child that places everything in her own closed world view. Here, the team wrote a completely new 3D to 2D projection software tool called the "water-drop perspective system" that forming a spherical projection of 3D objects around the subject of the animation, here filmed sequences of Tamás' young child. The wonder of the child exploring her environment is translated into a light and playful distortion of the shape and form of the environment.[19] Tamás Waliczky has completed two other works that subvert typical computer graphical systems, a circular perspective system in *The Forest*[20] and an inverse perspective system in *The Way*[21].

1.2.6 Deconstructivism in Interactive Art

"There is no longer any system of objects."
- Jean Baudrillard

Deconstructivism has its roots in postmodern theory, which, in one possible interpretation, states that any observed phenomenon is the result of a subjective projection into an illusion of meaning.[22] What one receives through the senses is assembled into a reality based on the assumptions and associations of the viewer. The viewer is an active agent in the environment, as there is no position from which one can form an objective understanding of truth and validity. Terry Eagleton, in the preface to his recent book, *The Illusions of Postmodernism*, sums up postmodern thought:

> "Postmodernity is a style of thought which is suspicious of classical notions of truth, reason, identity and objectivity, of the idea of universal progress of emancipation, of single frameworks, grand narratives or ultimate grounds of explanation. Against these Enlightenment norms, it sees the world as contingent, ungrounded, diverse, unstable, indeterminate, a set of disunified cultures or interpretations which breed a degree of skepticism about the objectivity of truth, history and norms, the givenness of natures and the coherence of identities."[23]

In contrast to the bottom-up process of constructivism within the modernist period, the process of inspection is a top-down endeavor. The viewer, given an impression of reality, must strip away the layers of associations and inferences that he or she contributes to the environment. As this "unlayering" continues, we arrive at the "pure reality", dissociating ourselves from assumed roles as viewers. This manner of inquiry is called *deconstructionism*, or the breaking down of a complex set of surfaces, or impressions of reality, in ever increasingly simplified terms. Therefore postmoderism replaces the notion of objects with the concept of process, where meaning is a subjective result of this interrogative process. The same principle applies to computer based interactive art. What is being presented to the subjective viewer are shadowy projections that are illusions of meaning. This notion of an aesthetically oriented surface of media is evoked by the French theorist Jean Baudrillard when he writes:

> "The description of this whole intimate universe – projective, imaginary and symbolic – still corresponded to the object's status as mirror of the subject, and that in turn to the imaginary depths of the mirror and 'scene'..... But today the scene and mirror no longer exist; instead, there is a screen and network. In place of the reflexive transcendence of mirror and scene, there is a non-reflecting surface, an immanent surface where operations unfold - the smooth *operational* surface of *communication*."[24] (italics added)

There are two striking keywords in this quote: *operations* and *communication*. The fact that these words have corresponding meanings between both postmodern theory and computer sciences is of particular interest. *Operations* intimate a feeling of a perpetual "unfolding" of process and, perhaps, allusions towards meaning, much like machine language op-codes (*op*eration *codes*) that evaluate to a recognizable computational meaning. Interactive media artist, Bill Seaman, describes this unfolding process within his writings on Re-embodied Intelligence.[25] I would like to re-interpret this statement to say that artistic expression is achieved through the *exposition of process*, whose formal description and implementation will be discussed in mathematical and technical detail later.

This forms a critical issue in interactive media art, as the concepts of subject, object, and the process of viewing become ever more blurred. If we consider the notion of an "interface" as a means to navigate thematic content, then there ought to be a similar coupling of the observer and being observed. Constructivist interactive works tend to place the viewer in the center of the constructed world, with dominant control of the environment. These modernist pieces, in my opinion, tend to keep control centralized at the user level, allowing the viewer to willfully engage and disengage from the piece. This notion of subjectivity and limited control in interactive environments have been previously explored by David Rokeby[26] and Perry Hoberman[27].

As described in later chapters in this thesis, I apply these deconstructivist techniques to derive salient characteristics from an interactive environment, such as motion, color, and form. The goal is to allow for a richness of experience, as the system, using parametric qualities of the real world to drive the interactivity, provides for an "unfolding" of process that both intrigues and surprises the audience.

Chapter

2 Review of Prior Interactive System Designs

As the formal study of interactive entertainment systems and their technological foundations is relatively new, there have been many disparate approaches to solving the issues that were discussed in Chapter 1. Each of the following projects, whose groupings I have taken the liberty of forming, offer a specific instance of a general category of interactivity. This chapter intends to provide a glancing introduction to a diverse body of both finished interactive works, their underlying technologies, and some of the current research foci in laboratories and institutions.

2.1 Interface Technologies

The challenges of successful interface design stem from the fact that the programmer/artist is creating a link between the viewer and the underlying computation. A shared representation must be formed so that there is a common semantic between the computer and the human. Figure 2-1 shows a block diagram of the significance of the interface. For example, the clarity of the desktop metaphor has made a large impact on the face of computing with the introduction of the Xerox Star and the Macintosh systems. The success stems from the commonality of the metaphor between the user and the machine. The user views the computer desktop as an extension of his/her daily life, borrowing on the associations of objects that occupy this space. The computer represents the desktop in familiar terms on its own, such as bitmaps, files, and tree hierarchies. While both parties have clearly different ways of organizing information, both the user and the computer can "come to terms" through this over-arching desktop metaphor.

As it is through the interface that the user expresses his or her intent, this shared metaphor forms a basis through which cause-and-effect relationships are established. Continuing with the desktop metaphor, the user quickly learns that dragging a file object into the trashcan is consistent with the expectation of functionality that is assigned to that action. That is to say that the iconic representation of the trashcan, within the shared metaphor of the digital desktop, intimates its own functionality. One would not expect to have a new file created, for example, upon the execution of this action. The expectation of a response given a particular action is directly proportional to the associations that the user takes into the environment, based on his/her prior experiences with similar objects. The more the expected action occurs, the more strongly the cause-effect association is reinforced.

Likewise, in interactive computer art, the set of interface objects, both physical and non-physical, directly influences the expectation of functionality. This can be both a benefit and a liability, depending on the intention of the artists and whether they wish to inherit the general associations of the interface that they use. What then occurs when objects are "overloaded" semantically speaking? In computer art, one tends to reuse familiar computational interface objects such as mice, keyboards, and monitors, especially in pure computer works, such as CD-ROMs and Virtual Reality. It is certainly odd for people to walk into a museum and see a set of computers and accept what is presented as a work of art, perhaps partly because of their daily associations with the computer as an "object". However there has been an increasing amount of research into new interface technologies and new interactive works

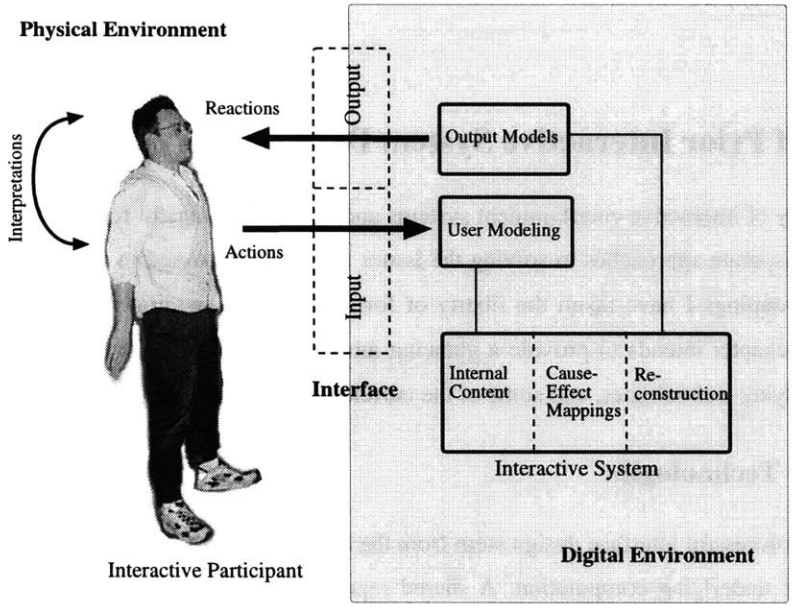

Figure 2-1. Interface as translator between the physical and digital worlds

that exploit these new offerings. With new interfaces, it is possible for artists to create unique cause-and-effect relationships between the user and the underlying system. The following sub-sections address several key groupings.

2.1.1 Image-based Interfaces

Machine vision is a difficult problem to overcome and has been a highly active focus for the past few decades in universities and corporate research groups. The goals of machine vision are to create scene descriptions based on its visual contents in order to enable further processing. Therefore computer vision is typically an interface to other computational components such as robot navigation, video coding[28], and 3D scene reconstruction[29]. The frustration in the field of vision interfaces is that although basic visual understanding is simple to us, the processes of having a machine perform a simple vision problem, such as object segmentation, is a monumental task. While computer vision is complicated in the general sense, there have been many successes in creating interactive systems with particular assumptions and constraints. Most of the success has been when only a low to mid-level understanding is desired. Table 2-1 divides manners of representations that break down into low-, mid-, and high-level scene understanding and shows some sample tasks that such a representation could enable. As is readily apparent, humans typically operate consciously on the highest level, further to the frustration of computer vision researchers.

At the lowest levels, David Rokeby, in his acclaimed work *Very Nervous System,* treats the vision input as a group of pixels.[30] Yasuaki Matsumoto, in his 1995 interactive installation *Schwerkraft und Gnade*, used computer vision techniques to track a viewer's torso and hands, creating pulsating graphical star objects at these points.[31] Mixed media artist Natalie Jeremienko uses a surveillance camera that continually watches the Golden Gate Bridge, performing vertical motion analysis in order to detect a person jumping off of the bridge.[32] These works do not attempt to achieve any understanding of the visual environment, only deriving certain characteristics from a real-time video stream.

Many advances in the field of mid-level computer vision have come from the Vision and Modeling Group (VisMod) at the M.I.T. Media Lab. This group has been successful in both the development of new vision understanding algorithms as well as a few artistic applications of this technology. One of the leaders in computer vision, Alexander Pentland, together with Baback Moghaddam, has created notable advances in face and facial expression recognition.[33] Christoher Wren has written *pFinder* which uses blob analysis to track arms, hands, legs, and torso of a single user within a static natural background.[34]

Knowledge Level	Image Representations	Enabled Tasks
Low-level	Pixels, small local-neighborhood	Background differencing, gradients
Mid-level	Edges, "blobs", PCA eigenvectors	Feature Tracking, classification of simple gestures
High-level	Objects	Scene understanding

Table 2-1. Three coarse levels of image understanding

Bruce Blumberg applied the *ALIVE* system[35], developed by the VisMod Group, to a virtual dog named "Silas" which is an autonomous agent that reacts to the commands of the human "owner." Together there is an element of play between the two characters in this scene: the user can pick up a virtual red ball on the screen and throw it, to have Silas retrieve it and drop it at the feet of the owner. This work has been featured at both SIGGRAPH'95 and the ARTEC'95 Media Art Biennale. Flavia Sparacino has used these vision tools to create several interactive works. *DanceSpace* allows both amateurs and professionals to create graphics shadows and musical compositions that are carved out through motion of the body. Her *Typographic Actor* uses *Media Creatures*, multimedia objects that exhibit autonomous behaviors based on state spaces that are driven by viewer actions. *DanceSpace* has been shown at the Symposium for Interactive Art at Connecticut College, 1997.[36]

2.1.2 Audio, Voice, Musical Performance Based Interfaces

As audio is a more simple 1D signal, in terms of compute complexity in comparison with 2D images, audio as an interface has been more widely used in artistic environments. Although this thesis will not directly deal with sound analysis and synthesis, it is nevertheless noteworthy to mention these research areas, as DSP is the common language between both disciplines.

Maja Spasova, in her 1995 installation *Sibyl*, used voice recognition as a interface to allow the viewer to navigate through a densely woven narrative state space.[37] While relying on an accurate speech recognition system, the piece used a state-space content representation where the viewer would merely make turns at discrete points in time.

More appropriately, William Oliver used Chirp-Z transformations and Cepstral analysis, based on the DSP work of Eric Metois[38], in *The Brain Opera*[39] installation, *The Singing Tree*, in order to compute pitch sustain and variation, brightness, volume, and format content of the user.[40] The system then uses Sharle, a computer music generation system written by John Yu[41], to harmonize around the sung pitch using a number of mapping modes based on the quality and stability of the singer. The singing participant is encouraged to explore different styles of singing, ranging from crisp and concise to random and off-key, in order to elicit a musical response from the

interactive system. Of all of the projects noted in this chapter, William Oliver's system design and implementation uses a model of interaction very similar to that which is proposed in this thesis.

Other performance oriented projects with the Opera of the Future Group at the M.I.T. Media Lab include Tod Machover's *Hyperstrings Trilogy*: *Begin Again Again* for hypercello, *Song of Penance* for hyperviola, and *Forever and Ever* for hyperviolin.[42] These works use a modified acoustic instrument that provide real-time data of performance technique to a computer. The computer measures certain features of the performers style, such as wrist action, bow placement, and bow pressure. This analysis is used to provide and manipulate real-time digital accompaniment to the live human performer.[43] The Opera of the Future has created several additional hyper-instruments for the *Brain Opera*, which will be discussed in greater detail in Chapter 7.

2.1.3 Tangible and Physical Interfaces

With traditional computer interfaces, the user is constrained to that which can be represented on the small monitor that is in front of him/her. Therefore, all visual attention is tightly concentrated towards this one object, making it difficult to create larger, more-inclusive environments, in my opinion. In order to bypass this problem of using computational interface methods, there has been an increasing amount of dedication into the investigation of using both commonplace and exotic objects as means for interactivity. This allows the artist to leverage off the context-specific associative values that we have towards such objects, further embedding artistic meaning within the environment.

Paul Sermon in his *Telematic Dreaming* work uses two beds as a teleconferencing arena where two remote participants visually communicate with each other while typically being removed at a large distance.[44] The use of the bed contextualizes the significance of the interactions between the two people, as the artist wishes to play with the notion of privacy, intimacy, and sexuality while, paradoxically, using technologies that are generally intended for the widespread broadcasting of television. While the video installation is not interactive in terms of computer-mediated art, it illustrates the strong associative powers that audience members have towards the interface object itself: the bed. In many cases, as this work was exhibited in public spaces, the audience members were uncomfortable in entering this normally private space with a remote stranger to the amusement of on-lookers.

Jeffrey Shaw, director of the Institute for Image Media at the Zentrum für Kunst und Medientechnologie (ZKM), used a modified stationary bicycle as an interface means through which the viewer navigates a computer graphics generated "city" of literary text.[45] The casual and playful use of this ordinary object, in conjunction with a virtual world, created an environment that thematically linked data exploration with a correspondingly exhaustive physical activity.

Brygg Ullmer and Hiroshi Ishii, both from the M.I.T. Media Lab, use phycons and phandles that redirect the computational desktop metaphor back into a set of physical objects.[46] In their *Tangible Desk* application, users manipulate these physical objects in order to navigate image and data content. The ease of the environment stems from the natural affordances and constraints that such objects inherently contain. For example, a sliding set of knobs are an intuitive physical instantiation of a graphical desktop slider or scaling command, as the limitations of the object itself, e.g. the slider has a fixed length, allow for an immediate understanding of the boundaries of the cause-

effect relationships. Furthermore, Hiroshi Ishii and his Tangible Media Group, have created the *Ambient Room* which uses physical objects, such as bottles and boxes, as containers that change the state of the environment, depending on whether they are opened or closed. Maggie Orth and Matt Gorbet use the symmetric and self-replicable qualities of geometric triangles with microprocessors in them to create a set of user-assignable construction kits.[47]

Joe Paradiso and Chris Verplaetse have created an augmented conducting baton that features three accelerometers, three pressure-sensitive pads, and a IR LED.[48] Teresa Marrin used this device as one of the hyper-instruments featured in the Brain Opera performance, giving musical directions to a series of sound samples that were dynamically called up.[49] In addition, Joe Paradiso has created a carpet that uses a grid of PVDF wire and two Doppler radar transmitter/receivers in order to create a sensor floor that has been used for musical performance.[50]

These above projects and research areas have stirred much interest in the field of interactive art. Hiroshi Ishii's and Joe Paradiso's works haven been invited to Ars Electronica and CHI'97, respectively, both of which have strong art and design conference themes. While, in my opinion, it is important to explore alternative interface designs, it is nevertheless a challenge to overcome any "novelty" with which a participant might view a work. This is to say that a work cannot solely rely on the "newness" of the interface design. Although Jeffrey Shaw's use of a bicycle is indeed interesting initially, its novelty quickly wears off as the VR environment that one navigates is not compelling enough to sustain any interest, in my opinion. This is due to the bicycle merely acting as an elaborate "joystick" offering no more degrees of freedom than what is available in current interface objects. Good interface design should not merely replace one input device (i.e. joystick) with another (i.e. a bicycle), but should strive to as many degrees of freedom that make "sense" in a particular environment.

Another difficulty with tangible interface design, in my opinion, is with the representation of the environment. For example, Hiroshi Ishii has created several bottles that "contain bits" that, when opened, release their contents into the ambient computational environment. The bottles suffer from the same problem as the state-space narratives that I described in Chapter 1 - the representation of the environment is reduced to a series of binary states, i.e. "bottle open" or "bottle closed". Although this is a personal suspicion, it is difficult to account for a range of qualities through binary encoding, as the number of bottles must increase in order to encode more and more states of the environment, creating a chaotic mess of physical objects that clutter a desktop. The best of both worlds would be to create physical objects that could yield a continuous quality, e.g. the angle of a box lid, making it easier to build more expressive systems.

2.2 Narrative and Script based Systems

One of the significant styles of interactive computer art has arisen from the application of cinematic aesthetics and rules to multimedia technologies. Glorianna Davenport, with her background in documentary filmmaking, has led the Interactive Cinema group at the M.I.T. Media Lab to the creation of several works that are based on narrative and documentary genres. The *Wheel of Life*, created in 1992-93, is a rich multimedia theatrical experience that immerses the viewer in a large constructed environment, using the three elements of Earth, Water, and Air as thematic basis for unifying the structure.[51] Current research into scripting systems is being performed by Stefan

Agamanolis through Isis, a Scheme-like multimedia scripting language that allows authors to quickly implement many interactive cinematic works.[52] This scripting language has been used by Freedom Baird to create *Sashay*, an interactive installation in progress that responds to a viewer's gestures to navigate through a dream narrative.[53] Mr. Agamanolis, together with Michael Bove, has used his own scripting system to build *Reflection of Presence*, an augmented teleconferencing system in which multiple participants can visually and aurally interact with one another.[54] In addition to this work, Isis has also been applied to the 1995 interactive movie, *Wallflower*, also by Stefan Agamanolis, Michael Bove, and Shawn Becker.[55] Claudio Pinhanez uses computer vision interface technologies to drive an interactive theater piece in which a performer can conduct, following a pre-written script, a choir of graphical creatures.[56]

2.3 Combinatorial Systems

Another often used approach to interactive systems design is through the use of Combinatorics, where a set of object primitives are available to the viewer to interconnect in order to create ever more intricate configurations.

The World Generator by Bill Seaman is, in my opinion, a clear example of a system of combinatorics, where Bill Seaman has coined the term "Re-embodied Intelligence" in order to describe its working process.[57] In this work Bill Seaman gives the audience a menu bar filled with graphically represented daily objects, such as chairs, that can be instantiated in a 3D virtual environment. Onto the objects the viewer can assign meaning through the selection of another menu selection of video texture maps and audio. In addition to these surfacial qualities, the objects can be assigned a set of behaviors that dynamically change the placement and characteristics of the instantiated scene objects. The audience can navigate the created world to watch the movie textures and hear the spatialized audio, forming a visual and sonic poem that surrounds the viewer. The more audience members are using the system, the denser the constructed world becomes, playing out the set of combinations that Bill Seaman provides.[58]

Another clear example of combinatorics is the *Triangles* project of Matt Gorbet and Maggie Orth. As described above, this work consists of a large number of identical physically triangles that can be attached to one another to form complex geometric shapes and surfaces. With each new triangle that is added to the system, the two "free" sides form a branching node in a tree structure. The total number of combinations depends of the number of assignable symbolic meaning that can be given to a single triangle. Overall, the combinatorics for four triangles with three symbolic means per triangle is astounding: 1620 unique configurations. One sample application, which is to be shown at the 1997 Ars Electronica Festival (Linz, Austria), is a user contribution system in which people can add replies to several daily questions that are prompted according to the current topological layout of the triangles.[59]

Chapter

3 Outline of Proposed Model

3.1 Overview

With both the technical and theoretical aspects of the problem of interactive art and entertainment in mind, this chapter serves as an outline of the author's model of interactivity.

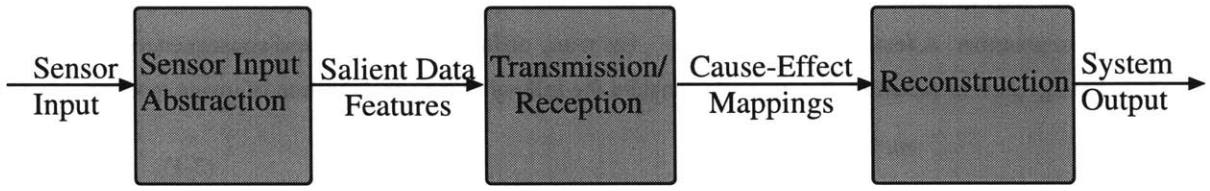

Figure 3-1. Block diagram of proposed model.

Shown in Figure 3-1 is a simple block diagram of the three main components that have been identified in this proposed model of interactive art. The *abstraction* component takes in one or more interface signals and deconstructs the interactive environment into a multi-dimensional, continuous element salient vector that summarizes the key aspects of the physical world. In the *transmission* sub-system, this salient vector is broadcasted to one or more remote sites, while a set of remote salient vectors, corresponding to remotely located participants, are received and mathematically integrated into one collective representation. Finally, this collective salient vector serves as a parameter set for an output system, that produces the cause-effect relationship between the viewer(s) and the system.

3.2 Salient features

According to my electronic Webster's, *salient* and *saliency* are defined as:

> *1. sa.lient \'sa--ly*nt, -le--*nt\ aj [L salient-, saliens, prp. of salire to leap - more at S]ALLY 1: moving by leaps or springs : JUMPING; specif : SALIENTIAN {a ~ amphibian} 2: jetting upward {~ fountain} 3a: projecting beyond a line, surface, or level : PROTUBERANT 3b: standing out conspicuously : PROMINENT, STRIKING {~ traits} - sa.lient.ly av*
>
> *sa.lience or sa.lien.cy \'sa--ly*n(t)s, -le--*n(t)s\ \-ly*n-se-, -le--*n-\ n 1: the quality or state of being salient 2: a striking point or feature : HIGHLIGHT*

This defines *saliency* as a quality of appearance, including a concept of a "feature". A feature here could be interpreted as an identifiable quality that sufficiently describes an object or event in a manner so as to summarize its appearance. When one uses features to communicate, precision is less of an issue than an impression that is held to be important by the observer/communicator. Thus a feature acts as a semantically higher-level representation of the said phenomenon while reducing the amount of description required to communicate its qualities to others.

Not coincidentally, "features" have a strong significance in many machine understanding applications such as image analysis, data compression, and classification systems.[60] In these applications, features are used to reduce the amount of data required to represent an observed state or quality that is present in the input sections of the system. Features are merely a mathematical approach for building higher levels of representations that are often used to reduce the computational load further down the system pipeline. For example, Giri Iyengar uses features derived from moving image sequences, such as motion, texture, luminance, and color to automatically annotate and index movies. In this case, movies are reduced to a few salient qualities that can be more easily accessed and searched.[61]

If features are higher-level representations qualities of input data, then what do they represent? Their meaning and significance are application specific and, thusly, depend on the associations of the researcher/programmer. A feature such as "motion", has many different definitions and consequences depending on the initial assignment of meaning. For example consider the following feature definition of image motion:

$$m_t = \sum_{i=0}^{N}\sum_{j=0}^{M}\left|I_{i,j,t} - I_{i,j,t-1}\right| \tag{3-1}$$

where I is an N x M matrix representing a monochromatic image at time t. This definition of motion is merely a summation of point-by-point illumination differences between two subsequent images. The more the image changes, in terms of pixel illumination, the higher the computed variable, m, becomes. However, it is easy arguable that such a feature is not appropriate, as a small translation of the same image, perhaps due to camera noise or tripod jitter between two frames, could produce a large change in this motion feature. Therefore, the use of derived features stems from a conceptual model that the researcher has in mind and the assumptions that are made a priori, for better or for worse. Either we have to make assumptions regarding the meaning of our feature m or further abstract its meaning so as to include higher levels of generality, such as motion merely meaning "changes in pixel illumination" rather than "movement of objects within the camera frame."

3.3 Salient Vector

Continuing with the above discussion, I wish to propose the concept of a *salient vector*, which is a vector that contains all of the salient features of an input signal. Although the dimensionality of the vector can vary from application to application, it is generally required that the vector length be constant within the system. The elements within the vector are the set of scalar values of the features that are generated from the input section of the interactive environment. Let us define an L-dimensional vector:

$$\Theta = (\Theta_1, \Theta_2, \Theta_3, \ldots, \Theta_L) \tag{3-2}$$

where each of the elements are scalar, continuous values that represent some computationally derived feature from the input. Taken as a whole, this grouping allows us to form a *salient vector* that groups the entire set of features into one mathematical expression. This is useful for notational reasons, as we will perform subsequent mathematical operations on this salient vector.

As this is a vector quantity, vector operations are mathematically valid and well defined. For example we can sum, differentiate, and perform simple statistical operations on this vector notation. Furthermore, I will discuss

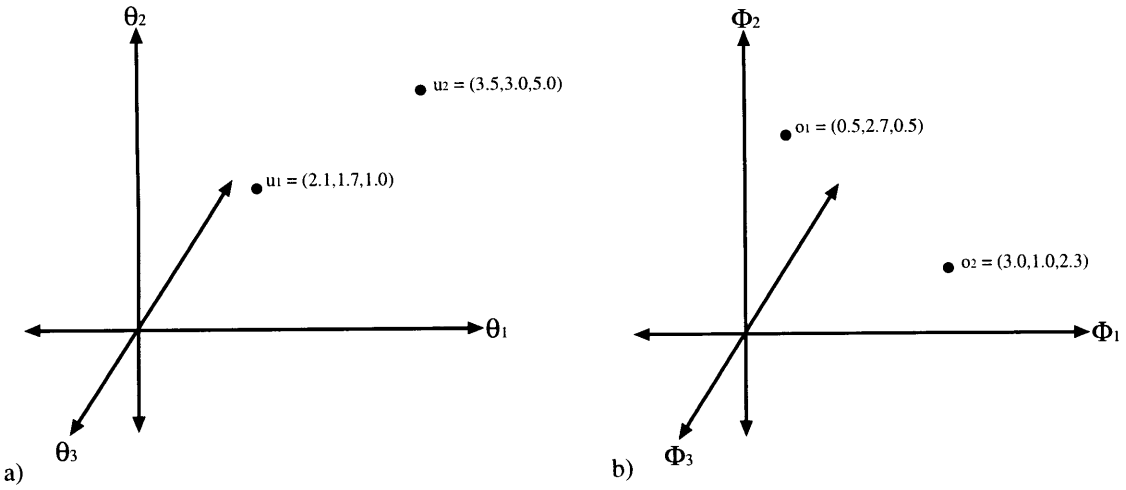

Figure 3-2. a) Two user representations in a three-dimensional salient space. b) Two output representations in a three-dimensional output parameter space.

the possibilities of projecting, or mapping, this vector onto spaces of different dimensionalities. Therefore averaging two salient vectors can be simply stated as:

$$\bar{\Theta} = \frac{\Theta_1 + \Theta_2}{2} \tag{3-3}$$

Furthermore, a weighted averaging of N salient vectors, whose significance will be explored later in Chapter 5, can be expressed as:

$$\bar{\Theta} = \frac{\sum_{i=0}^{N} \alpha_i \Theta_i}{\sum_{i=0}^{N} \alpha_i} \tag{3-4}$$

where the weighting multiplier α is a vector some the same length of Θ and contains the corresponding weights to be used. Note that we are performing element-by-element operations here as each individual element within the vector has a unique semantic meaning from the input data.

What makes salient vectors so important to the field of interactive media are their mathematical robustness. As will be presented later in this thesis, through Chapters 4 to 6, interactive systems are, more or less, a series of mappings that transform input sensor data to an observable output. Since the salient vector is both continuous and orthogonal, with proper interface implementation, it may be possible to consider a mathematical *salient space* that is created from the basis functions of the salient vector. Figure 3-2 shows a sample salient space with three axes that are formed from three hypothetical input sensors. Every point in this space should be defined at least in mathematical terms. The importance of a salient space is that it is the complete and exhaustive set of all interactive inputs that are possible within the environment. Likewise, there is an output reconstruction parameter space that is the set of all possible outputs from the system, where each axis corresponds to one of the model parameters of the reconstruction. Every point in this parameter reconstruction space should be mathematically defined and represents one possible output. Figure 3-3 shows a sample output parameter space.

While it may be conceptually interesting to formally express all of the possible interactive input and output as mathematical spaces, this concept is not intended to be philosophically profound but rather to define what is meant by cause-effect mapping. The entire proposed model is based on the notion that *parametric-based interactive computer art stems from a series of mathematical space transformations that map from one salient space into another.* There can be either one mapping designed into a system, i.e. input salient vector to output reconstruction model, or many that follow one another. While there can be as many intermediate mappings as desired by the artist/engineer, it always begins with the input salient space and ends with the output parameter space. Figure 3-3 shows an arbitrary mapping from the salient input space to output parameter space. It is my conjecture that through these space transformations, the interactive viewer forms cause-effect relationships between his/her actions and the systems reactions. This is due to the fact that the system is continuously performing the salient deconstruction of the physical environment, into the salient vector, performing intermediate transformations, and then applying these results to a reconstruction model.

With this overview of salient features, vectors, and spaces, we can investigate in more detail how to create these mappings. Chapter 4 concentrates on techniques of data abstraction, through interface technologies, into a salient vector. Chapter 5 introduces how multiple interactive participants can be accommodated within such a system. Chapter 6 presents several sample output reconstruction models that provide for the final mapping of the salient vector into an observable alteration in the interactive environment. Six implementations of this system, as part of Tod Machover's *Brain Opera,* are detailed in Chapter 7, serving as illustrations of how an entire environment can operate in a real-world setting.

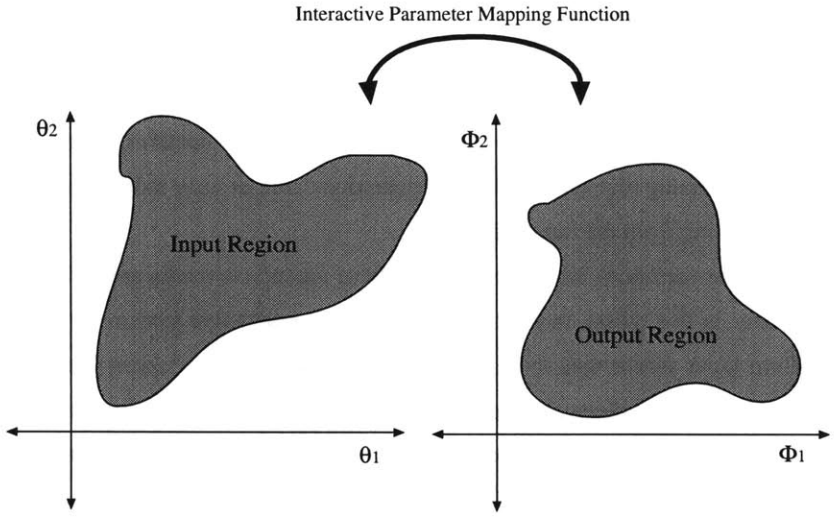

Figure 3-3. Arbitrary example mapping between salient input and output parameter space.

Chapter

4 Abstraction

"Take away the sensations of them; let not the eyes see light or colours, nor the ears hear sounds; let the palate not taste, nor the nose smell, and all colours, tastes, odours and sounds, as they are such particular ideas, vanish and cease, and are reduced to their causes, i.e. bulk, figure and motion of parts."

- John Locke, "Essay concerning Human Understanding"

This chapter describes in detail the first component of the proposed model: the use of sensing technologies in order to derive a salient vector that represents one or more qualities of the physical world, including the interactive viewer. Whereas input technologies allow the computer to "see" and "hear" the real world, machines can only react to those things that are apparent at its input. Clearly, the system can only form interactions based on the choice of interface objects and the types of actions they elicit.

As presented in Chapter 1, my intention within the field of interactive computer based art is to create a deconstructivist system that can derive a number of salient impressions from the real world and reapply these qualities into a series of cause-effect mappings. With this in mind, the interface becomes the point of decomposition where complex 'realities' are broken down into very simple terms, e.g. human shape and form are broken down into a matrix of pixels through a camera. Such an inherent property of an interface is important for the communication of abstract qualities due to the fact that such a deconstructed representation can be arbitrarily reapplied to other systems. This reapplication of computationally derived qualities could be considered a cross-coupling of saliency, where one set of qualities is echoed within another. As an example, consider a simple photoelectric sensor which maps light intensity to a continuous voltage output that again serves as a control parameter to an amplifier playing some sound. This system arbitrarily maps one quality of the environment, i.e. the overall lighting, to another quality, i.e. sound volume. Although the cross-coupling of light and sound may not be the most compelling application, it does illustrate that abstracted qualities can be easy remapped to output phenomenon, especially when these qualities are mathematically continuous in nature. The reader is referred to George Legrady's work for additional examples of image analysis within media art.[62]

This chapter is dedicated to investigation of a range of sensing technologies that are available, what they actually measure, which qualities can be derived from them, and, most importantly, several possible abstracted salient vectors that can be formed. Many of the following sensing technologies are currently being researched by many others and are not unique to this thesis project. In particular, I decided to focus on vision-based interfaces as there is a wide range of qualities that can be mathematically derived from the input. Many of these algorithms have been implemented in my aforementioned interactive work, *What Will Remain of These?* (1997), and have proven themselves to be reliable and robust. Additional user abstraction possibilities will be presented in Chapter's 7 and 8, which are more focused within the context of *The Brain Opera* project.

4.1 Vision Input

Researchers have been using computer vision as input into learning and adaptive systems for a few decades. It is an obvious choice to investigate vision because humans use vision as their primary sense. While humans have an inherent capacity to use vision for scene understanding, machines are not quite so fortunate. As mentioned in Chapter 3 and reinforced by leading computer vision researcher Ted Adelson, computer vision is very difficult and research is still in creating mid-level vision systems.[63]

The following chapter sections examine a few qualities that can be computationally derived from a set of input images. These approaches yield only low-mid-level analysis, mapping an image into a set of features that can be used as a salient vector to describe the images in a highly reduced form. It is important to notice that there is no notion of scene understanding here, as this would require substantially higher-order analysis and constraints.

In order to define the notation, all images are from a stream of digital images, sampled at discrete time intervals from time 0 to time interval T and listed a set of images $\mathbf{I} = (\mathbf{I}_0, \mathbf{I}_1, \mathbf{I}_2, \mathbf{I}_3, ..., \mathbf{I}_T)$. Images are considered to be N x M matrices, where N is the number of pixel rows and M is the number of pixel columns of the images. The independent variables x and y are used as indices refering to a single pixel in the image matrix. The indexing of the matrix begins at the upper left corner, exactly as it is performed in matrix notation.

4.1.1 Texture Analysis

Common definitions of texture analysis use spatial derivatives in order to provide information about the changes between neighboring pixels in an image. The assumption here is that texture is a high-frequency component in the image signal. If one looks at a rug or tree bark, one can see large amounts of illumination variance over the surface of the object. Figure 4-1 shows fours examples of image texture.

Local texture could be defined as a gradient analysis that is performed on a pixel-by-pixel basis. Let us consider the following texture function definition, using function I to represent a continuous two dimensional input signal:

$$t(I(x,y)) = \nabla I(x,y) = \begin{bmatrix} \frac{\delta I(x,y)}{\delta x} \\ \frac{\delta I(x,y)}{\delta y} \end{bmatrix} \tag{4-1}$$

Texture then is considered a gradient function of the source image, which is a partial derivative in both x and y. However, as images are discrete entities, one has to define the step size over which the derivative is taken, which is

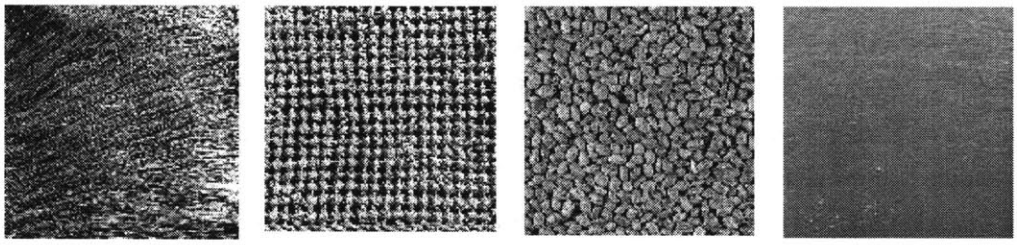

Figure 4-1. Four examples of image texture, from left to right: fabric1, fabric2, food, water

of considerable significance. The step size relates to the expected frequency range of the original source image, subject to aliasing should this be too large. As a starting point, let us consider one initial definition of gradients within discrete images, given a position within the image at location (x, y):

$$z[x,y] = \nabla \mathbf{I}_{x,y} = \begin{bmatrix} \mathbf{I}_{x,y} - \mathbf{I}_{x+\Delta x,y} \\ \mathbf{I}_{x,y} - \mathbf{I}_{x,y+\Delta y} \end{bmatrix} \tag{4-2}$$

where Δx and Δy are pre-defined step size constants that are set by the algorithm. If we set Δx and Δy to 1, then we subtract spatially neighboring pixels that are immediately to the right and above from the current pixel. Note that the gradient for each pixel yields a two-dimensional column vector, $\mathbf{z}$, one gradient in the x direction and one in the y direction. If we perform this gradient calculation on every pixel in the N x M source image, we can concatenate all of the texture vectors into one texture matrix $\mathbf{Z}$ of the dimensionality 2 x (N * M):

$$\mathbf{Z} = \begin{bmatrix} \frac{\delta \mathbf{I}_{0,0}}{\delta x} & \frac{\delta \mathbf{I}_{0,1}}{\delta x} & \frac{\delta \mathbf{I}_{0,2}}{\delta x} & \cdots & \frac{\delta \mathbf{I}_{N-1,M-1}}{\delta x} \\ \frac{\delta \mathbf{I}_{0,0}}{\delta y} & \frac{\delta \mathbf{I}_{0,1}}{\delta y} & \frac{\delta \mathbf{I}_{0,2}}{\delta y} & \cdots & \frac{\delta \mathbf{I}_{N-1,M-1}}{\delta y} \end{bmatrix} \tag{4-3}$$

Note that each successive column of this matrix is the gradient vector of each pixel of the source image $\mathbf{I}$, scanning left-to-right, top-to-bottom. Thus we can refer to the matrix as:

$$\mathbf{z} = \begin{bmatrix} \mathbf{z}_0 & \mathbf{z}_1 & \mathbf{z}_2 & \cdots & \mathbf{z}_{N*M-1} \end{bmatrix} \tag{4-4}$$

Although, contrary without intentions, we end up with a larger data structure than with what we started. Therefore we should consider methods to summarize these qualities. Once we have defined this matrix $\mathbf{Z}$ in such a manner, we can perform simple statistical operations on it. We can calculate the expected mean of the image texture function of the image:

$$\bar{\mathbf{z}} = E[\mathbf{Z}] = \frac{\sum_{i=0}^{N} \mathbf{Z}_j}{N*M} \tag{4-5}$$

This produces a mean vector of dimensionality two, where each element of the column vector is the expected mean of the texture gradient in both x and y. We can also calculate the covariance matrix from the texture matrix and the expected mean:

$$\mathbf{K}_\mathbf{Z} = E\left[(\mathbf{Z} - \bar{\mathbf{z}})(\mathbf{Z} - \bar{\mathbf{z}})^T \right] \tag{4-6}$$

The elements in the covariance matrix are the second central moments of the texture vector components. The diagonal elements $k_{0,0}$ $k_{1,1}$ are the variances of the distribution of each gradient which we are most interested in. These mean and variance measurements, as they are continuous values that describe a salient aspect of the image, can be used as components of a salient vector that wishes to represent such qualities.

4.1.2 Color Analysis

Another useful quality in vision is chrominance statistics, as color is one of the most basic tools for artists. A short formal discussion of color spaces is needed before we continue with the analysis stages. Typically there are four common color spaces in use which are referred to as the RGB, CMY, YIQ, and HSV color models. The RGB color space is the most straightforward to understand as it is an additive color space. Colors are simply linear combinations of three prime colors: red, green, and blue. Many commercial video cameras and most digitizers produce signals that encode the RGB components of incoming light.

The second important color model, CMY, is a subtractive color space as the encoded colors, cyan, magenta, and yellow are the color complements of red, green, and blue. In this space, the color white is at the origin and filters are used to subtract away color from white. The relationship between RGB and CMY color spaces is merely:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} C \\ M \\ Y \end{bmatrix} \tag{4-7}$$

The third color space is the YIQ color model which is used in the NTSC (National Television Standards Committee) of transmitted video signals. The YIQ color space uses one component, Y, to encode the luminance of an image and the I and Q components to encode the chromanance signal. To convert a color signal from RGB to YIQ space, a 3 x 3 transformation matrix is used:

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.528 & 0.311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{4-8}$$

As you can see, it is easy to move from one color space to the other with the help of these color models. However, the last color space, and the most useful to this section on color is the HSB (hue, saturation, and brightness) color space which is based on the artist's innate knowledge of tint, shade, and tone. The B axis encodes the "brightness", the H axis (which is radial) encodes the "hue", and the S axis encodes the "saturation" of the color. Thus this color space is useful as hues that would be described as similar are spatially near one another, whose significance will be investigated hereafter. Therefore, we will use the HSB space as the default color space for our color analysis.

Unfortunately there is no elegant and generalized transformation from RGB space to HSB space, like the ones listed above. The conversion from RGB encoded colors to the HSB space is done algorithmically, whose pseudo-source code is listed in [64].

If we perform the same statistical mean and variance calculations on the color representation of the video image, as was done in Eqs. 4-5 and 4-6, a salient color vector can be derived. The average color values will be contained in the means, while the variance will act as an indicator of how similar or different the color content is. Images that have a smooth color consistency will have small variance values. This, of course, only produces meaningful values if the underlying color distribution is indeed Gaussian.

4.1.3 Form Analysis

Once we have segmented a video image into foreground and background elements as described in [65], we can begin to look at computational methods to examine the form of the input. The term form is used here loosely and can refer to a number of possible interpretations relating to shape. In this section, we will consider one possible definition of "shape" which is most related to the concept of moments of mass. This approach was taken in 1996 during my internship at the Mitsubishi Electric Research Laboratory in Cambridge.[66] It is assumed that we have already successfully isolated the foreground object(s) from the background, which is rarely the case in real-world situations. But, in order to simplify the following discussion, I will consider only the ideal case.

Consider an image of a person standing in a specific pose. We can say that the shape of his/her body determines the form of the foreground object. With the a priori knowledge that the foreground objects are indeed people, we can form an internal model that attempts to account for the distribution of image mass. As the overall shape of the body could be approximated through a series of attached image blobs, it makes sense to use image moments. The zero-th order, m_{00}, and two first order two-dimensional moments, m_{01} and m_{10}, can be written as Riemann integrals:

$$\begin{aligned} m_{00} &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I(x,y)\,dxdy \\ m_{01} &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} yI(x,y)\,dxdy \\ m_{10} &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} xI(x,y)\,dxdy \end{aligned} \tag{4-9}$$

To derive the center of mass from these values:

$$\bar{x} = \frac{m_{10}}{m_{00}} \qquad \bar{y} = \frac{m_{01}}{m_{00}} \tag{4-10}$$

However, the center of image mass is of limited interest to us. If we also compute the second moments:

$$\begin{aligned} a &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^2 I(x,y)\,dxdy \\ b &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} xyI(x,y)\,dxdy \\ c &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} y^2 I(x,y)\,dxdy \end{aligned} \tag{4-11}$$

To find the orientation and size of the image mass, we need to find the eigenvectors and eigenvalues of the following matrix:

$$\begin{bmatrix} a & b \\ b & c \end{bmatrix} \tag{4-12}$$

In this case the eigenvectors represent the orientation of the mass while the eigenvalues are relative measurements of the amount of spatial variance that exists along these principle axes of image mass. Furthermore, it is possible - although not described in this thesis document but can be found in [67] - to perform a series of recursive image moment analyses that will yield a pair of eigenvectors and eigenvalues for each level of an octree. Taken as a whole, this produces a finer description of the shape of an articulated body.

4.1.4 Motion Analysis

Figure 4-2. Nude Descending a Staircase by Marcel Duchamp

One of the key questions of video sequences is how objects move within the frame over time. Artists have used the study and simulation of motion for a long time, as it is necessary to allude to a moving object in a static art form.[68] Figure 4-2 shows a well-known work by Duchamp of a figure walking down a staircase. Here the temporal behavior of the subject is made clear although the painting is frozen in time. In terms of mathematics, the question is, given a set of video frames, how individual image elements move over time. Again the key question is at what semantic level do we wish to describe the notion of 'elements'? If we are working at the lowest pixel level, the analysis becomes one active area of research known as optical flow techniques. The sub-section will review the techniques discussed in the Bergen paper.[69] Other substantial work can be found in [70].

Given two image frames $\mathbf{I}[t]$ and $\mathbf{I}[t\text{-}1]$, we can form both a translation and affine transformation model. This model has the following assumption:

$$\mathbf{I}_{x,y}[t] = \mathbf{I}_{x-p_x,y-p_y}[t-1] \tag{4-13}$$

which is to say that the image is the same except for a translation at every point by a variable quantity (p_x, p_y). In order to calculate these quantities, we take a least-squared error approach and try to minimize the error quantity. Using the mathematical discourse as described in Bergen et al., we end up with the set of following equations:

$$\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \end{bmatrix} = -\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix} \tag{4-14}$$

Solving for p_x and p_y will yield the motion change in the x and y coordinates. This above equation assumes that motion is modeled by simple translation. However, it is possible to view motion as an affine transformation which requires six parameters that describe the motion:

$$p_x(x,y) = a_x + b_x x + c_x y \tag{4-15}$$

$$p_y(x,y) = a_y + b_y x + c_y y \tag{4-16}$$

which, when the set of derivatives with respect to each parameter is set to zero, produces a system of six equations:

$$\begin{bmatrix} \sum I_x^2 & \sum xI_x^2 & \sum yI_x^2 & \sum I_x I_y & \sum xI_x I_y & \sum yI_x I_y \\ \sum xI_x^2 & \sum x^2 I_x^2 & \sum xyI_x^2 & \sum xI_x I_y & \sum x^2 I_x I_y & \sum xyI_x I_y \\ \sum yI_x^2 & \sum xyI_x^2 & \sum y2I_x^2 & \sum yI_x I_y & \sum xyI_x I_y & \sum y^2 I_x I_y \\ \sum I_x I_y & \sum xI_x I_y & \sum yI_x I_y & \sum I_y^2 & \sum xI_y^2 & \sum yI_y^2 \\ \sum xI_x I_y & \sum x^2 I_x I_y & \sum xyI_x I_y & \sum xI_y^2 & \sum x^2 I_y^2 & \sum xyI_y y^2 \\ \sum yI_x I_y & \sum xyI_x I_y & \sum y^2 I_x I_y & \sum yI_y^2 & \sum xyI_y^2 & \sum y^2 I_y^2 \end{bmatrix} \begin{bmatrix} a_x \\ b_x \\ c_x \\ a_y \\ b_y \\ c_y \end{bmatrix} = -\begin{bmatrix} \sum I_x I_t \\ \sum xI_x I_t \\ \sum yI_x I_t \\ \sum I_y I_t \\ \sum xI_y I_t \\ \sum yI_y I_t \end{bmatrix} \tag{4-17}$$

If this is solved for the six parameters, then they can be substituted in Eqs. 4-15 and 4-16 to solve for p_x and p_y. In terms of compute complexity, Eqs. 4-15 and 4-16 are far more demanding compared to Eq. 4-14, which is an important issue when we discuss real-time implementation issues.

The output of optical flow analysis produces a $N * M$ flow field, whose elements have a Δx and Δy component. This flow field estimates how each pixel is moving from one frame to another and can be used to derive motion qualities from a scene. Figure 4-3 shows a few subsequent video frames and the optical flow field - using the translation only model - that they produce. The flow field is visualized here as a series of lines that indicate the direction and amplitude of the estimate pixel motion between frames. The summation window size was 5 x 5.

The authors in Bergen et al. propose the use of a Gaussian pyramid to break down the original image sequence iteratively by powers of two. With the Gaussian pyramid, a coarser description of the scene motion is produced, since high frequency visual textures will be gradually filtered out as one goes higher in the pyramid levels. Furthermore, less calculations will be required because for each level we go higher in the pyramid, the total number of pixels in the image drops by a factor of four. Therefore, if we choose to perform optical flow at the Gaussian pyramid level three with an original NTSC monochrome video image of size 640 x 480 (307200 pixels), we will only have optical flow dimensions of 80 x 60 (4800 pixels). This yields a compute savings of 64 times in the calculation of the optical flow! Furthermore the optical flow will be far more robust to both camera noise and high-frequency texture information.

If we wish to summarize the motion within a moving video sequence it is possible to perform a few additional operations. Consider a set of optical flow matrices, (**O**[0], **O**[1], **O**[2],.... **O**[T]), of size N x M, that are formed by the above technique over an entire set of T frames that represent a scene. It would be possible to perform a weighted average of the entire set of optical flow matrices:

Figure 4-3. Three frames from a video sequence and their corresponding optical flow.

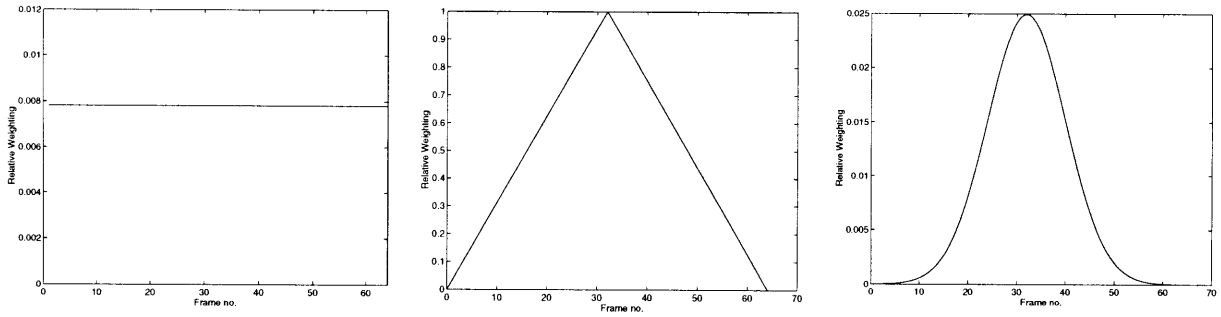

Figure 4-4. Three weighting functions for optical flow integration: linear, triangular, and Gaussian.

$$\overline{O} = \frac{\sum_{t=0}^{T-1}\sum_{i=0}^{N-1}\sum_{j=0}^{M-1}\beta(t)O_{i,j}[t]}{\sum_{t=0}^{T-1}\beta(t)} \tag{4-18}$$

where $\beta(t)$ is a weighting function associated with time step t. Figure 4-4 shows two possible weighting functions.

4.2 Fish

A Fish is an capacitive sensing device that measures the amount of body mass is within a electric field, closely related to the famous Theremin musical instrument developed by Leo Theremin in 1920. The idea is that these sensing devices are able to capture hand gestures non-intrusively, where the participant does not need to be "wired" to any computers directly. This "tetherless" interface is appealing because there are no distractions from unsightly cables, allowing the physical space to be open in which the sensor operates. There are two modes in which the Fish can operate: shunt or transmit. In the shunt mode, which is used by the Theremin, the user grounds an electric field that is created between the transmitter and receivers. Conversely, in the transmit mode, the participant is the actually transmitter, creating an electric field between the hand of the performer and the receivers. The transmit mode is used by the *Sensor Chair* and the *Gesture Wall* - one of the thesis projects - as described in Section 7-6-1.

The Fish sensor consists of a small hardware box, four receivers, one transmitter, and a RS-232 serial link to a computer. Through the use of four receivers, it is possible to derive hand coordinates of the user in 3D space. As the transmissive properties depend on body weight and height, it is required to calibrate the Fish system for each person. In order to derive reliable 3D coordinates, the system must be "software calibrated", whereby each participant must place their hands at particular fixed locations within the sensing space. With these known spatial coordinates and their corresponding Fish outputs, it is possible to perform a linear least-squares fit in order to generate a matrix transformation that maps from the Fish output space to 3D hand coordinate space.

The Fish sensor has been integrated into several performance environments, including Penn & Teller's performance of Tod Machover's piece *Media Medium*[71], David Waxman's *Gesture Fames*[72], a collaboration with the artist formally known as Prince, and the aforementioned components of *The Brain Opera.* A more detailed description of the Fish sensor can be found in [73].

Chapter

5 Transmission

One of the key features of the digital environment is in its interconnectivity. Computers and other computational devices - either 'smart' or 'dumb' - can be connected to one another, forming intricate data "topologies". This allows for a computational 'reality' that spans large physical displacements, where environments are able to be continuous despite of the obvious real-world discontinuities.

The recent explosion of network computers and development tools with interconnectivity in mind has also peaked the interest of multimedia artists in two different aspects. First, the Internet forms a solid base for low-to-no cost distribution of artistic "product." Second, communities of audiences are spontaneously formed through the infrastructure of the Internet, much like the physical roads of yesteryear that paved smaller communities into a larger collection. Given that there are both are new means to distribute content as well as a "live" community of audience members, it is possible to investigate different genres of interactive art that leverage off of these two new affordances.

In many ways, these new Internet-based works are reminiscent of public art works, generally using simple metaphors that bridge a large number of their daily lives. As the Internet is rapidly becoming a standard means of communication and building community, any artistic expression that is placed on the Internet, generally through the World Wide Web, *is* indeed public. Many works have been made to take advantage of this community such as Ken Goldberg's charming work, *The Telegarden*, in which people, over the World Wide Web, plant and maintain *real* plant seeds that grow and flourish based on the care of the remotely located viewers.[74]

5.1 Models of Transmission

Figure 5-1 shows a block diagram of the transmission component of this interactive system. The purpose of this section of the system accomplishes four major functions. First, the salient vector may or may not be mapped into a transmission salient vector, which is described later in more detail. Second, the transmission salient vector that is derived from the abstraction sub-system described in Chapter 4 is broadcast to all other machines on the network. Third, each local machine receives one or more remote salient vectors. Last, each machine remaps the remote salient vector, if necessary, into the local interactive salient vector space and performs mathematical operations to "integrate" the remote vectors in some well defined manner. The word integrate is not necessarily intended to denote the mathematical operation of integration, but rather a loose denotation of combining all of the salient vectors that are at its disposal. This last operation is the most fundamental, as it is our interest to 'fold' in as many different people as possible into one environment. As the reconstruction sub-system of the interactive environment - described in Chapter 6 - is local to each installation environment, we need to instill the sensation in the local participant that the environment is populated – or even haunted! – by several unseen people that are given a manifestation over the network.

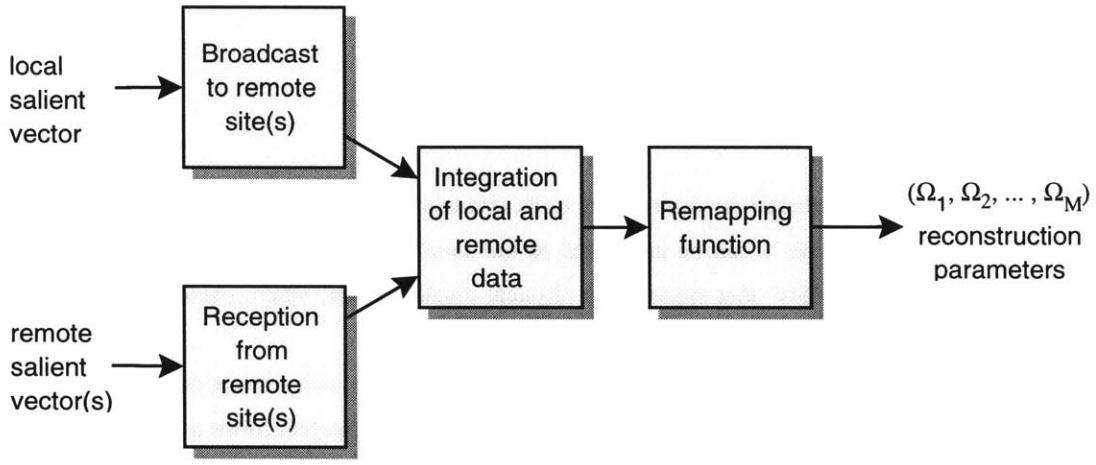

Figure 5-1. Overview of the transmission sub-system of model.

In the discussion – and implementation – of this model of interactive system design, I consider each computer as being a local computational host for one interactive environment. This means that rather than just raw computers being connected together, interactive environments are the basis of the network, bringing together a small society of viewers. As this is a higher-level view of data communication, it merely means that the network becomes part of the interactive experience. The *connectivity* of the digital world is manifested by the ability of the user to sense the presence of another viewer who is having a similar experience within a different environment. Due to the fact that both people are within this shared space, we can discuss forms of creating both digital shadows of the others as well as creating larger representations of a community of viewers/participants. We do not wish to make the viewer conscious of the interconnected technology, which is already very familiar to people (i.e. phones, ATM's), but rather to make the participant have the sensation that they are part of a seamless, continuous digital environment. Such a goal, to me, is very humanistic, bringing together a large community of people that is sharing a common experience. This experience can be trivial or profound, but a sense of community can only arise out of the knowledge that people's lives, like the underlying computer hardware and software, are interconnected.

This above goal is similar to Fiona Raby's & Anthony Dunne's piece *Fields and Thresholds* in which two remotely placed benches serve as the input/output device. When one person sits on one bench, heaters underneath the bench at the remote end warm up, communicating a subtle sensation of an immaterial presence that is conveyed through the data network. This simple, yet highly effective, use of data network is still capable of transmitting an essence of being to remote locations.[75]

5.2 General Communication Issues

When using data networks – and shared environments in particular - there are a few properties that need to be addressed by the artist/engineer. In particular, there is an latency involved in any data communication. Latency here is the amount of time that transpires between the transmission and reception of the data. There are a few different causes for the latency. Normally, there is some time required for the interactive system to make the operating system calls to prepare a buffer for output. Second, many I/O systems use data queuing that waits for a certain amount of

data to be transmitted before initiating the actual transfer, in order to optimize for "burst" transfers. Third, there is a delay in the actual transmission in the transport layer of the data communication. The Internet consists of a web of connected machine that link one computer to another over a series of intermediate steps, each with a corresponding delay. Fourth, there may also be data queues at the input port at the remote machine that adds an additional delay. Finally, there is also a slight delay incurred during the operating system calls on the remote machine to read the data at the communication port into a usable private data buffer. Taken together this total latency may be on the order of several hundred of milliseconds.

As we are forming a representation of the user at discrete moments in time, this group delay causes the receiving machine to have an "out-dated" view of the remotely located participant. This can be a problem with very responsive interactive environments, where this delay can give a feeling of a disturbing asynchronicity between the two people. In fact, computer networked games are particularly vulnerable to this update rate, giving certain players a slight advantage based on their network connection group delay.

Another major variable in network communications is data bandwidth that can be steadily supplied between two points on a network. In works that use the Internet as the transmission medium, it is difficult to guarantee a bandwidth at any point in time. Since the IP protocol is designed to act as a shared resource, available bandwidth for any one program is dependant on how many people are using the Internet at the same time, particularly if others sending large amounts of data. Certain protocol layers on top of IP, such as Multicast, try to be more efficient in the distribution of data to multiple receivers. The reader is referred to [76] for more information on real-time concerns of Internet broadcasting.

In order to design an effective shared interactive environment, we need to address other non-hardware and software issues. Most importantly, as each location is representing both the local and remote participants, the cause-effect mappings become more abstract and, perhaps, more complicated for the local viewer to understand. The local viewer will always ask: *What is my contribution? What am I affecting versus the other people?* These questions can be addressed by how the artist designs the hierarchy of interactivity, whether the local and remote participants have equal "weight" and "influence" in the environment or whether the local viewer will have more importance than the remote people. This "weighting" of participation will be more formally discussed later in this chapter.

What then differentiates a shared environment in which multiple people can engage in a meaningful dialog from a mere playground where everyone engages in a self-absorbed monologue? This is the challenge in designing multi-user installations and whose proposed methods will be discussed here.

5.3 Salient Vector Transmission and Reception

In order to create a shared interactive environment, we first must form a representation of a collection of participants. This gives us two representations: one of the local participant and one of the entire group, each of which will be used as a basis for interactions. The communication aspect is simply to send a representation of the user, based on the salient vector, to each of the remote machines. Usually this vector is the same as the one that is produce from the abstraction section as outlined in Chatper 4. However, it may be useful to perform additional data abstractions depending on the application and the subsequent weighting of local and remote participants.

For example, consider an optical flow scenario as described in Chapter 4, where we are producing a dense two-dimensional representation of the user based on his/her movements in an image plane. If we keep these as a floating-point (4 bytes) representation over a large image, then the salient vector tends to be large and perhaps contains more information than we need. In this case we could consider sending a lower-resolution version of the optical flow information, using a higher level of the image pyramid, or even sending only the average of all of the flow vectors. In this case we want to reduce the overall bandwidth of the user representation from one point to the other. The critical question to ask is, *do we need to send all information in order to represent the local viewer?* In some cases this will be yes, at other times no, and is dependent upon the interactive scenario.

Once the user representation of a participant at a remote site is received, it may have to be reverse transformed into the local user salient representation. If we consider the earlier example of the reduced resolution optical flow example, we would need to inverse the compression portion. The optical flow was at a higher Gaussian pyramid level, we would need to upsample the data in order to produce enough data points to match the local user respresentation. This could simply be a tri-linear interpolation for the number of in-between data samples. Figure 5-2 shows an example of this upsampling.

However, in general, the remote user salient vector will be of the same format as the local salient vector. This simplifies the system somewhat as we can directly perform subsequent operations immediately, without any intermediate mapping functions. This is required in order to make the subsequently detailed mathematics simpler.

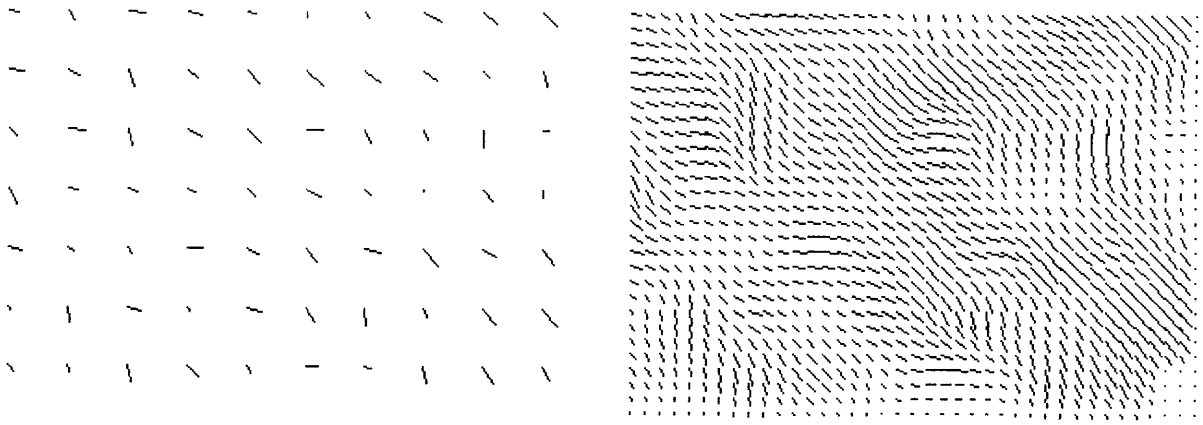

Figure 5-2. Using tri-linear interpolation to upsample, by a factor of 16, an optical flow transmission salient vector.

5.4 Operations Performed Upon Community Vectors

Once we have the local and the remote user representations, we can now consider how to create a single "group identity" salient vector. These can range from the very simple, i.e. averaging, to more abstract combinations. As the salient vectors are the same length, we can define a set of vector operations that will create a unified salient vector from this community. At times we will want this unification operation to be either an even combination of all of the participants in the shared environment, other times we will want to weight the local user more heavily than the

remote participants. This is in accordance to the artists wishes to provide adequate cause-and-effect cues to the local viewer, as the more complicated and profound the mathematics become, the more frustrating the experience can be.

The net effect should be to give the impression of an *inhabited* space where the environment is reactive to more than the immediate local viewer. Should the viewers cooperate with each other? Should the interactive system be sensitive to how similar or how different the participants are from one another? What is the relationship between the local viewer and the set of remote people? The next few chapter subsections propose a few types of mathematical operations that could be considered by the artist/engineer given these vectors. This is only meant to be a sampling and is in no ways complete. The methods implemented in the thesis project are outlined and evaluated in Chapter 8.

5.4.1 Statistics

The most apparent step to unify the set of local and global viewer representations is to perform simple vector statistics, such as mean and variance. In order to define the nomenclature for the rest of the chapter, let us define the local viewer salient D dimensional vector as:

$$\mathbf{u}_0 = (u_0, u_1, u_2, \ldots, u_{D-1}) \tag{5-1}$$

and the N remote vectors as a N x D matrix:

$$\mathbf{R} = \begin{bmatrix} \leftarrow & \mathbf{r}_1 & \rightarrow \\ \leftarrow & \mathbf{r}_2 & \rightarrow \\ \leftarrow & \mathbf{r}_3 & \rightarrow \\ & \vdots & \\ \leftarrow & \mathbf{r}_N & \rightarrow \end{bmatrix} \tag{5-2}$$

where the set of N $\mathbf{r}$ vectors are of the same data types as the local user salient vector $\mathbf{u}$ and represent the salient vector that was received from one of the N remotely located machines. If we consider these two different sets, let us define a N+1 x D comprehensive user matrix:

$$\mathbf{U} = \begin{bmatrix} \leftarrow & \mathbf{u}_0 & \rightarrow \\ \leftarrow & \mathbf{u}_1 = \mathbf{r}_1 & \rightarrow \\ \leftarrow & \mathbf{u}_2 = \mathbf{r}_2 & \rightarrow \\ & \vdots & \\ \leftarrow & \mathbf{u}_N = \mathbf{r}_N & \rightarrow \end{bmatrix} \tag{5-3}$$

This matrix contains all of the user representations into once simple format. From here we can calculate the mean user vector:

$$\overline{\mathbf{u}} = \frac{\sum_{i=0}^{N} \mathbf{u}_i}{N+1} \tag{5-4}$$

which weights the contribution of everyone equally. However we may wish to weight everyone's contribution differently:

$$\bar{\mathbf{u}} = \frac{\sum_{i=0}^{N} a_i \mathbf{u}_i}{\sum_{i=0}^{N} a_i} \tag{5-5}$$

where the set of coefficients a are the relative amount of influence that each local/remote viewer has in the environment. This allows particular interactive participants to control the environment more than others, lending a sense of hierarchy to the work. Note that this influence does not have to be a constant coefficient if we define the weighting coefficient to be a function of the interactive environment:

$$\bar{\mathbf{u}} = \frac{\sum_{i=0}^{N} a(\mathbf{U}, i)\mathbf{u}_i}{\sum_{i=0}^{N} a(\mathbf{U}, i)} \tag{5-6}$$

Possible uses for the weighting function could be that highly active viewers are given a reward of a proportionally higher weighting. Also this weighting could be a simple function of time, where each participant is given a "window of control". Consider if we used the following periodic weighting functions:

$$a_i[t] = 0.5 * \cos(\frac{2t\pi}{T} + \frac{2i\pi}{N+1}) + 0.5 \qquad \text{i} = 0....\text{N} \tag{5-7}$$

where T is the period of the cycle, i is the user, N is the total number of remote users. If we consider 4 interactive viewers being linked to the same environment, we get weightings that are shown in Figure 5-3. As you can see, the weighting, or control of the environment, oscillates from one user to the other. Although everyone is making a contribution, he/she who is given more weighting varies steadily over time. Of course, more complicated assignment of control could be considered, although the artist/engineer runs into the difficulty of making the relationship between the viewer and his/her corresponding effect ever more abstract and difficult to understand.

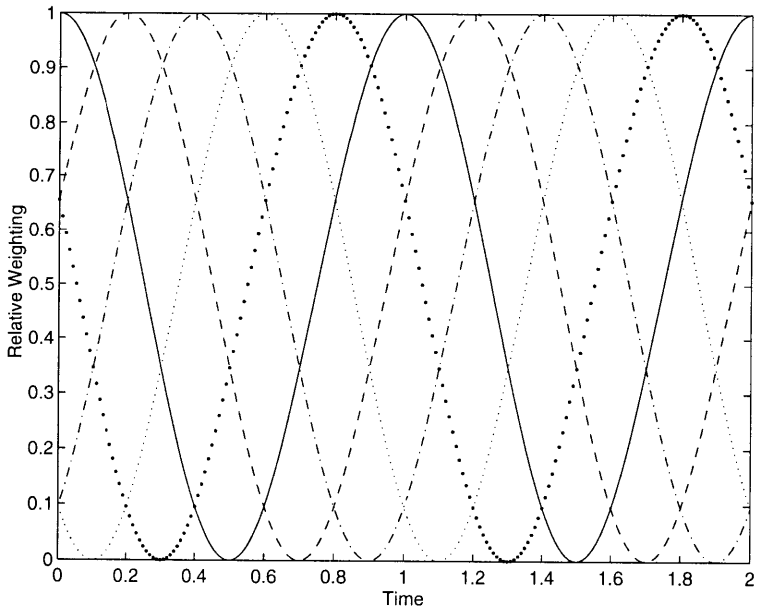

Figure 5-3. Example weighting functions for 5 interactive participants.

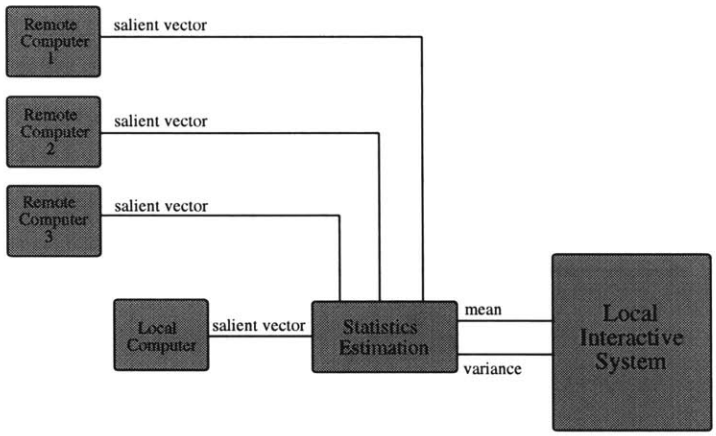

Figure 5-4. Using mean and variance estimations to drive an interactive environment.

With this averaging completed, it would be useful to derive other statistics from the user representation such as variance between each of the elements in the salient vector. In other words, the variance of the salient vectors is a rough indication of how similar or different the qualities of the participants are. When the representation of the interactive users are numerically close and clustered around the "average" salient vector, this indicates that the viewers are qualitatively similar. What "similar" means in this usage depends on the application. If we have an installation that uses a Fish sensor to derive a salient vector of motion characteristics, a low variance is indicative of similarity of first-derivative motions.

To calculate the estimation of the covariance matrix:

$$\mathbf{K} = \mathrm{E}[(\mathbf{u} - \overline{\mathbf{u}})^{\mathrm{T}}(\mathbf{u} - \overline{\mathbf{u}})] \tag{5-8}$$

The resulting D x D (where D is the dimensionality of the salient vector) matrix contains the variances of the salient vector elements along its diagonal and are of the most interest to us. Let us then create a single variance vector that contains only the diagonal elements:

$$\boldsymbol{\sigma} = (k_{0,0}, k_{1,1}, k_{2,2}, .., k_{D,D}) \tag{5-9}$$

With these mean and variance vectors, we can make our interactive environment respond to these measurements rather than the direct user representations. Luckily, this variance vector is also of length D, the same as each salient vector, making it easier to integrate into the system. Such a system could be described as in Figure 5-4. Furthermore, it may prove useful to do second-order statistics that calculate the mean and variances of these first-order statistics over time. This will give the system an impression of how organized the interactive participants are over blocks of time, rather than at a single instant of time. We could accumulate, over time, a block of T mean and variances:

$$\overline{\mathbf{U}} = \begin{bmatrix} \leftarrow & \mathbf{u}_0 & \rightarrow \\ \leftarrow & \mathbf{u}_1 & \rightarrow \\ \leftarrow & \mathbf{u}_2 & \rightarrow \\ & \vdots & \\ \leftarrow & \mathbf{u}_T & \rightarrow \end{bmatrix} \qquad \boldsymbol{\Lambda} = \begin{bmatrix} \leftarrow & \boldsymbol{\sigma}_0 & \rightarrow \\ \leftarrow & \boldsymbol{\sigma}_1 & \rightarrow \\ \leftarrow & \boldsymbol{\sigma}_2 & \rightarrow \\ & \vdots & \\ \leftarrow & \boldsymbol{\sigma}_T & \rightarrow \end{bmatrix} \tag{5-10}$$

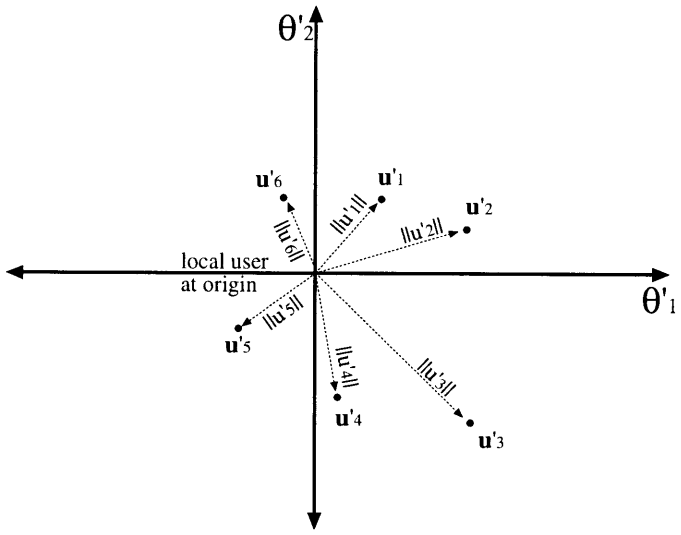

Figure 5-5. Remote salient vectors oriented around local participant.

where each row of the matrices correspond to a given mean and variance at each time step. Clearly we can take a mean and variance measurement of the mean matrix and of the variance matrix. Such a second-order statistic are useful to measure changes of characteristics over time, as each row in these matrices are a function of time. If the variances of these second-order statistics are low, this indicates that the first-order statistics are very consistent over the time slice.

5.4.2 Differential Estimation

While the above section introduced a formal mathematical approach on how to estimate the "similarity" of a group of interactive participants, it may be equally as useful to determine how each individual differs from one another. We will define the term "to differ" as a distance function that represents how far apart two salient user vectors are. As each salient user vector can be considered as a point in a *D*-space, where *D* is the number of dimensions of the salient vector, we can consider several possible distance metrics between two vectors, **x** and **y**. Figure 5-5 shows a two dimensional salient vector space and six remote salient vectors within that space, translated so that the local viewer is at the origin. The most popular distance metric is called the Euclidean distance:

$$d(x,y) = \|x-y\| = \left[\sum_{i=1}^{D}(x_i - y_i)^2\right]^{\frac{1}{2}} \tag{5-11}$$

Another distance function is the maximum value:

$$d(x,y) = \max_i(|x_i - y_i|) \tag{5-12}$$

and the absolute value distance:

$$d(x,y) = \sum_{i=1}^{D}|x_i - y_i| \tag{5-13}$$

Let us consider a scenario where we wish to create a numerical equivalence of how different the remote users are from the local user. Let us define an intermediate difference vector that is just the vector between the local user **u** and a remote user $\mathbf{r}_i$:

$$\mathbf{d}_i = \mathbf{r}_i - \mathbf{u} \tag{5-14}$$

We could then define a N x D difference matrix D that contain all of the differences between the local and remote users:

$$\mathbf{D} = \begin{bmatrix} \leftarrow & \mathbf{d}_1 & \rightarrow \\ \leftarrow & \mathbf{d}_2 & \rightarrow \\ \leftarrow & \mathbf{d}_3 & \rightarrow \\ & \vdots & \\ \leftarrow & \mathbf{d}_N & \rightarrow \end{bmatrix} \tag{5-15}$$

We could visualize this matrix as a set of vectors that originate from the local user, as done in Figure 5-5. The more similar the remote users are with the local user, the more they cluster around the origin. Conversely, the less similar they are, the more scattered the visualization will be.

5.5 Formation of Interactive communities

This chapter's goal has been to create an interactive community that extends a virtual environment out towards a number of physically remote participants. As the Internet provides for a "smooth" transmission of real-time data, with some constraints, it is possible to think of a set of interactive environments as one unified, but distributed, environment. Therefore, the participants will be co-interacting with both the thematic content of the work and each other. Since the system formally creates representations and models of a collection of users, the notion of a community becomes a central part of the thematic content of the work.

Through the salient vector, the operations needed to define a set metrics for community are relatively straightforward. Had we chosen to use a non-continuous description of a user, i.e. using classification, such high-level analysis would have proved very difficult due to the non-linearity of the representation. In my opinion, this is one of the main arguments for such a model of interactivity that this thesis proposes: the ease of mathematical operations that can create simple mappings from one space to another. This notion builds upon the work of Judith Donath, who formally researches the aspects of a community of users on digital networks. Her work "Visual Who" visualizes the association of every user in a community to a number of conceptual "anchors" such as his/her group affiliation or membership to the softball team. The underlying system that performs the visualization is a set of spring forces that virtually attach each user's name to a selectable set of anchor points. The stronger the association is at any point in time, the visualization shows the changing of the spring equilibrium point, corresponding with the motion of the user's name throughout the space.[77]

Similarly, we can, through the use of statistical analysis of local and remote users' salient vector, visualize the relationships between each other. We can bring the similarities of each other to the surface or we can concentrate on how different all of the participants are. This is very different from the majority of interactive environments in which the system only responds to the local interactions of the singular viewer. While the sensation of being part of

an interactive digital community may be unfamiliar – and correspondingly abstract - at first, gradually, as our lives increasingly become dependent on such digital pathways, such modes of communication will hopefully become second-nature.

One of the design challenges in creating collaborative interactive environments is whether it is necessary to give absolute frames of reference to the local user. This situation reminds me a little of the Prisoner's Dilemma, where two isolated persons must "interact" without any form of absolute communication, receiving only indirect information that is a function of both of their actions as a feedback mechanism. For example, if an installation is using a motion as its interface and the collaborative environment is responding to the similarity in motion, i.e. the system rewards cooperation between the two, then how does each person form a communication language to coordinate their actions? There is only the reconstruction of the system, as described in Chapter 6, that is given as a feedback. But the reconstruction is a function of the interactions of the participants and is being continually altered! So without any absolute channel of communication, i.e. a means through which symbols are not distorted between the input and output ends, it becomes a process to find means of communication *through* the interactive environment rather than *above*. While perhaps frustrating to many, it is of great intellectual and artistic interest to balance objective knowledge and subjective experience in order to make the interactivity more engaging. If too much control is given to the viewer, then it is my belief that the experience will quickly loose its appeal. Conversely, should a work be too abstruse then it will most likely frustrate and discourage the audience.

Chapter

6 Reconstruction

6.1 Recontextualization

The reconstruction section of this model of interactivity is responsible for producing the response component of the cause-effect relationship between the user and the environment, closing the feedback loop between the viewer and the content. Without this output feedback, it would be difficult for the user to recognize their significance in the environment, causing a breakdown of the interactivity.

Therefore the reconstruction section recontextualizes the presence and intent of the viewer in relation to the thematic content of the work and makes him or her an active participant. If one could think of an interactive work as a mirror, then we need to create corresponding "reflections" that this mirror gives back to us. Through the recontextualization of the user, we are shown an element of ourselves that may not have been apparent earlier. The objectivity of the computational system can provide additional insights into our behaviors, like a microscope removed from the immediate context. In some ways, this double inspection is ironic; we watch a system that is watching ourselves. In fact Monika Fleischmann's interactive work *Narcissus* uses this self-absorption as a corresponding allegory in reference to the tale of Narcissus.[78]

How the artist wishes to form output representations of the interactive viewer is the subject of this chapter. There are a wide range of styles possible that decode the system, ranging from an immediate atavistic correspondence such as in Myron Krueger's works[79], to a faint "whisper" of legibility, as in David Rokeby's *Very Nervous System*. Furthermore there are two major sets of variables here, 1) how the input parameters are mapped to output parameters and 2) how the output parameters are visualized. Given that the viewer is searching for reflections of themselves in the system, we can, as artists, either immediately fulfill this need to recognition or either delay this arrival point or deny it completely. The mapping and reconstruction style the artist chooses plays an important significance to the user being able to sense their presence. It is difficult to draw solid conclusions on which method is preferred, as different viewers will have different philosophical pre-dispositions on how literal their presence should be.

For example, two interactive installations of mine, *Winds that Wash the Seas* (1995)[80] and *What Will Remain of These?* (1997)[81], place themselves at opposite ends of the spectrum in terms of cause-effect mappings and revisualization. The *Winds that Wash the Seas* interactive system, based on the viewer's water-stirring and air blowing, directly maps these stimuli into spatial image warping and alpha-blending algorithms. For this work the cause-effect relationship is immediately apparent to the user. The visual and sound material is directly manipulated by the viewer, thus the viewer quickly "learns" the installation and can use these rules to navigate the content. However the later work, *What Will Remain of These?,* uses a more aggregate cause-effect interface relationship and abstract visual output. In this work, the system does not respond to just one viewer but whole groups of members, slowly "learning" the motion behaviors of the audience based on visual input from an array of surveillance cameras. As both the cause-effect mappings and revisualization are conceptual and abstract in nature, it is very difficult to

understand the rules of the installation. Thus the installation inverts the notion of learning, as it is the function of the digital environment to form a mid-level "understanding" (to use the word lightly) of the physical world rather than, as typical in interactive environments, the viewer learning the system.

These two interactive works elicit strongly different opinions from viewers. Some people have preferred the meditative simplicity of the earlier work while others have been bored by the direct nature of the mappings. With the later work, some viewers are frustrated by the lack of a direct cause-effect mapping, leading some to think that the installation is "broken", not aware that the mappings are accumulative over time. However, others have been intrigued by the kinetic abstract beauty of the installation output and they have revisited the work several times, eventually forming a cursory "understanding" of what was happening. Such a diverse set of reactions to these works appear to validate the assumption that the theory of fulfillment and denial in interactive installations warrants further experimentation.

6.2 Computer Graphics

The most frequently used reconstruction model takes computer generated imagery as its form of output, using either computer monitors or data projectors to place the output into the environment. In computer graphics, all of the structural content of the scene and either all or much of the surface is entirely computationally generated. One cross-over between computer graphics and image processing techniques, as discussed below, is with texture mapping that can "wallpaper" a set of polygons with either a still or moving image. However, let us assume that all visual output content is entirely synthesized and do not have immediate sources from the real visual world.

Virtual Reality, for example, is based on the use of computer graphics in order to create visual content dynamically that is based on the interactions of the viewer. Additional output hardware, such as VR goggles, are also occasionally used. Typically speaking, 3D scene rendering technologies are used that place the viewer within a computer generated scene, such as Jeffrey Shaw's *The Legible City* or Bill Seaman's *The World Generator.* However, several significant works, such as Toshio Iwai's *Musical Insects*[82] and Scott Snibbe's *Motion Phone*[83], use 2D spite animation as output. Interactive systems use of computer graphics most likely stems from the technology's economical success in the computer industry, as several hardware and software packages are available to provide high-performance levels.

Let us start with a very basic 2D graphical primitive, the circle. The form of a circle can be described with a three parameters, its (x, y) coordinate location of its center and the size of its radius. Thus we can say that a very reconstruction model be an instantiation of a circle with a single vector that encodes these three parameters:

$$\mathbf{c} = [c_x \quad c_y \quad c_z] \tag{6-1}$$

This vector is a three-dimensional vector and could be considered to form a circle space and each point in this space describes a unique circle. Therefore in order to link this representation into the interactive system, we need to describe a function **f** that maps points from our salient space into this three dimensional circle space, or in more mathematical terms:

$$f : \Re^N \rightarrow \Re^3 \tag{6-2}$$

where N is the number of dimensions of our salient space.

For sake of simplifying the discussion, let us assume we already have a 3 dimensional salient space that was generated through the abstraction sub-system of the interactive environment. In this sample scenario, we use a vision system to track a persons hand over a plane and estimate the diameter of the hand region. Therefore our salient vector would also be from a three dimension space:

$$\mathbf{s} = [h_x \quad h_y \quad h_d] \tag{6-3}$$

So, we need to define a set of functions that map from this 3-space to the reconstruction 3-space, which is normalized for our output screen :

$$f_{c_x}(h_x) = s_w * h_x * 0.5 + o_x \tag{6-4}$$

$$f_{c_y}(h_y) = s_h * h_y * 0.5 + o_y \tag{6-5}$$

$$f_{c_r}(h_d) = h_d * 0.5 \tag{6-6}$$

where s_w and s_h are screen width and height, in pixels, and o_x and o_y are origin offsets into the screen coordinates of the output monitor. This can also be made into a simple homogeneous matrix scaling and translation transformation:

$$\begin{bmatrix} 0.5*s_w & 0 & 0 & o_x \\ 0 & 0.5*s_h & 0 & o_y \\ 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} h_x \\ h_y \\ h_d \\ 1 \end{bmatrix} = \begin{bmatrix} c_x \\ c_y \\ c_r \\ 0 \end{bmatrix} \tag{6-7}$$

This output parameter vector **c** can then be passed into the 2D graphics rendering section of the system and shown to the interactive viewer. In this scenario, the interaction would be such that a circle follows the hand, changing size based on the apparent size of the hand, either due to wrist rotation or distance from the camera. This example is solely meant to illustrate one possible transformation from a salient vector to an output parameter vector. However, through the description of the thesis projects in Chapter 7, a more comprehensive examination of computer graphics is outside of the scope of this document and the reader is referred to [84] for more information.

6.3 Image Processing

A second method of reconstruction is with interactive image processing, which dramatically differs from computer graphics in that the output is based on either live or pre-recorded "real" (rather than purely synthesized imagery). Changes in the visual output can be accommodated through several possible image processing algorithms that alter the appearance and quality of the live or stored imagery. Here the presence of the user can be through the live imagery and/or the control parameters to the transformations.

The techniques that are introduced in this sub-section will be put into practice in Chapters 7 and 8. In the *Gesture Walls* and *Melody Easels* installations of *The Brain Opera*, these algorithms are placed into a real-time interactive environment, whose evaluations and conclusions will be presented in Chapter 9.

Figure 6-1. Two original images and an even alpha-blending

6.3.1 Alpha-blending

One common image processing algorithm is to "blend" together two images, where two images are averaged together, pixel-by-pixel to form an output in which both of the sources are visible. Figure 6-1 shows two source images and a resulting blending where both images have equal averaging weight in the output. A formal mathematical representation of this operation is:

$$\mathbf{I}_{i,j} = (1.0-\alpha) * \mathbf{I}^0_{i,j} + \alpha * \mathbf{I}^1_{i,j} \quad \text{i} = 0...\text{N}-1, \text{j} = 0...\text{M}-1 \tag{6-8}$$

The blending parameter α is the weighting function that is between 0.0 and 1.0. The lower the parameter value, the more image $\mathbf{I}^0$ is weighted in, and, conversely, the higher alpha is the more image $\mathbf{I}^1$ is weighted in. Note that α is constant here with the significance that this blending is uniform across the entire image and is not a function of time.

A classical blending algorithm is that of a cross-fade, where the tail of one video scene is faded into the beginning of another scene, making the transition "cut" less sharp. This example is one where the weighting functions between the two images is a function of time and must be calculated frame by frame. Consider a case where we have two video sequences $\mathbf{I}^0$ and $\mathbf{I}^1$, each 30 frames in length. To perform a cross-fade from $\mathbf{I}^0$ to $\mathbf{I}^1$ over this period:

$$\mathbf{I}_{i,j}[t] = (1.0-\frac{t}{30.0})\mathbf{I}^0_{i,j}[t] + \frac{t}{30.0}\mathbf{I}^1_{i,j}[t] \qquad \text{i} = 0...\text{N}-1, \text{j} = 0...\text{M}-1, \text{t} = 0...29 \tag{6-9}$$

The only difference between Eq. 6-8 and Eq. 6-9 is that we have a time-series set of images and an equation for the weighting function α [t]:

$$\alpha[t] = \frac{t}{T} \tag{6-10}$$

Where T is the number of images in the time-series. However Eq. 6-9 is still constant in terms of spatial screen coordinates. It would also be possible to define a weighting variable that is a function of i, j, and t, i.e. α [i,j,t], that will blend together pixels at different weights at different spatio-temporal portions of the image. For example, the

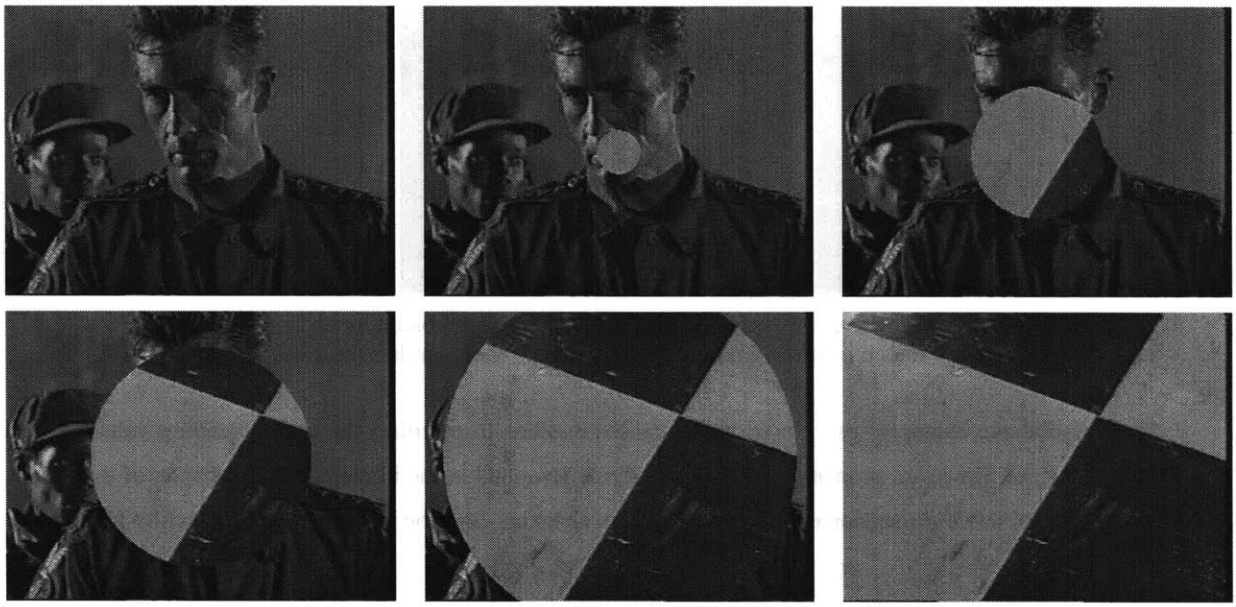

Figure 6-2. Using a time-varying alpha-blending matrix to transition between two video sequences.

film/video "wipe" effect, does a cross-fade blend but in a directional manner over time, such as left-to-right. However, we can create more novel types of weighting functions:

$$\alpha[i,j,t] = \begin{cases} 1.0 & \frac{\sqrt{(i-c_y)^2+(j-c_x)^2}}{\beta} < \frac{t}{T} \\ 0.0 & \frac{\sqrt{(i-c_y)^2+(j-c_x)^2}}{\beta} \geq \frac{t}{T} \end{cases} \tag{6-11}$$

where

$$\beta = \sqrt{c_y^{\,2} + c_x^{\,2}} \tag{6-12}$$

and c_x and c_y are the spatial coordinates of the center of the screen. Using this weighting function over T frames will cause a circle wipe to grow from the center. Transitioning between the two frames is pictured in Figure 6-2.

Other novel blending effects could be accommodated with a weighting function such as:

$$\alpha[i,j,t] = \sin\left(\frac{2tF_t\pi}{T} - \frac{2rF_s}{\beta}\pi\right) * 0.5 + 0.5 \tag{6-13}$$

where

$$r = \sqrt{(i-c_y)^2+(j-c_x)^2} \tag{6-14}$$

and F_t is the temporal frequency and F_s is the spatial frequency of the effect, in Hertz, and are assumed to be constants in this case. This will produce a wave-like cross fade that emanates from the center of the screen. A few frames from this visual transformation is shown in Figure 6-3 at different time points, t.

Figure 6-3. Using a radial "wave" as the source to an alpha-blending algorithm. The center of the screen is the origin of the effect, radiating outwards over time.

These above examples use closed analytical expressions to calculate the alpha blending values for each point in the images. However, in many cases, such as those described in the implementation chapter of this thesis, it is better to allocate a two-dimensional alpha matrix whose elements describe the weights on a pixel-by-pixel basis:

$$A = \begin{bmatrix} \alpha_{0,0} & \alpha_{0,1} & \alpha_{0,2} & .... & \alpha_{0,M-1} \\ \alpha_{1,0} & \alpha_{1,1} & \alpha_{1,2} & .... & \alpha_{1,M-1} \\ \alpha_{2,0} & \alpha_{2,1} & \alpha_{2,2} & .... & \alpha_{2,M-1} \\ : & : & : & & : \\ \alpha_{N-1,0} & \alpha_{N-1,1} & \alpha_{N-1,2} & .... & \alpha_{N-1,M-1} \end{bmatrix} \quad (6\text{-}15)$$

This could obviously be considered the solution set of the equation $\alpha[t]$ at each of the spatial coordinates in the time-varying image. Eq. 6-15 is usually a function of t and will be referred to as $A[t]$. Therefore to calculate a composite image at each time step:

$$\mathbf{I}_{i,j}[t] = (1.0 - \mathbf{A}_{i,j}[t])\,\mathbf{I}^{0}_{i,j}[t] + \mathbf{A}_{i,j}[t]\,\mathbf{I}^{1}_{i,j}[t] \qquad i = 0...N-1 \quad j = 0...M-1 \quad (6\text{-}16)$$

We can update the weights either on every frame or at arbitrary times, depending on the application. Chapter 7 will demonstrate a scenario, within *The Brain Opera* implementation, where the weight matrix has both continual alterations as a function of time as well as changes based on user events in the interactive environment.

6.3.2 Spatial re-mapping

Another widely employed image processing operation is from spatial re-mapping. This transforms the spatial coordinate system of an image such that it yields a "warped" version of the original. The impression that such an effect can produce is similar to a fun-house mirror, that stretches a viewer's image according the to bending of the mirrored surface. Other common occurrences of this effect is with standard video effect boxes that texture maps a video signal to a moving 3D planar surface. In an extended version of the same effect, virtual reality systems use texture mapping to "place" a video sequence over one or more polygons in 3D space.

Formally this processes takes a set of x and y coordinates of an original image and transforms them into another set of coordinates. Consider the following simple transformation:

$$[x \quad y]\begin{bmatrix}0.5 & 0\\ 0 & 0.5\end{bmatrix}=[x' \quad y'] \tag{6-17}$$

which simply scales the coordinate system down by a factor of two. If this transformation is performed over the entire set of x, y coordinates of an original image, the transformed image will be the same as the original but at a half the size along the x and y axes. It is important to note that add and even x's and y's will map to the same integer portion of the transformed region, e.g. when $x = 2$ and $x = 3$, we will get 1.0 and 1.5 respectively. As image indices are always whole numbers, we will need to low-pass filter the transformed image and then subsample in order to avoid aliasing.

It is possible to use a more general affine transformation of the coordinate space in order to accommodate other image warps that include image shear, rotation, and translation. Affine transformations do **not** model perspective alterations or warping of images over a sphere, for example. The reader is referred to work by Steve Mann for perspective transformations of images.[85] The equations for this affine transformation are:

$$\mathbf{x}' = a_x + b_x\mathbf{x} + c_x\mathbf{y} \tag{6-18}$$

$$\mathbf{y}' = a_y + b_y\mathbf{x} + c_y\mathbf{y} \tag{6-19}$$

This transformation requires six parameters. a_x and a_x describe the amount of translation and b_x, c_x, b_y, and c_y describe the amount of scaling and shearing. This can also be written as a matrix transformation:

$$[x \quad y \quad 1]\begin{bmatrix}b_x & b_y & 0\\ c_x & c_y & 0\\ a_x & a_y & 1\end{bmatrix}=[x' \quad y' \quad 1] \tag{6-20}$$

The inverse of the transformation matrix can be solved analytically as and substituted in the above equation. It is easier to illustrate the inverse mapping if we break down the affine transformations into translation and rotation/sheer operations:

$$\begin{bmatrix}x\\ y\end{bmatrix}=\begin{bmatrix}b_x & c_x\\ b_y & c_y\end{bmatrix}^{-1}\left(\begin{bmatrix}x'\\ y'\end{bmatrix}-\begin{bmatrix}a_x\\ a_y\end{bmatrix}\right) \tag{6-21}$$

The inverse of this 2x2 matrix is easy and is listed in [86].The final inverse affine transformation is thus:

$$x = \frac{a_y c_x - a_x c_y + c_y x' - c_x y'}{b_x c_y - b_y c_x} \tag{6-22}$$

$$y = \frac{-a_y b_x + a_x b_y - b_y x' + b_x y'}{b_x c_y - b_y c_x} \tag{6-23}$$

Using these equations it is relatively easy to render a transformed image, using the same number of machine calculations as the forward transformation if we lightly re-arrange these inverse functions to:

$$x = \alpha_x + \beta_x x' + \gamma_x y' \tag{6-24}$$

$$y = \alpha_y + \beta_y x' + \gamma_y y' \tag{6-25}$$

where the constants need only to be computed once and are:

$$\alpha_x = \frac{a_y c_x - a_x c_y}{b_x c_y - b_y c_x} \qquad \beta_x = \frac{c_y}{b_x c_y - b_y c_x} \qquad \gamma_x = \frac{-c_x}{b_x c_y - b_y c_x} \tag{6-26}$$

$$\alpha_y = \frac{-a_y b_x + a_x b_y}{b_x c_y - b_y c_x} \qquad \beta_y = \frac{-b_y}{b_x c_y - b_y c_x} \qquad \gamma_y = \frac{b_x}{b_x c_y - b_y c_x} \tag{6-27}$$

The reason why we care so much about the inverse affine transformation is because the spatial coordinates of the output image are known and fixed. With these equations we find the answer to the following question: "For each pixel in the *output* image, what is its corresponding pixel in the original?" Furthermore, as we know the a_x, a_y, b_x, b_y, c_x, and c_y of the affine transformation, we have everything we need to render this image.

However, solving this inverse transformation for each pixel of the output image we will get mappings to fractional indices of the original image. For example, output pixel coordinate (10,10) may correspond to pixel coordinate of (20.5, 30.1) of the original. There are several methods to solve this issue, each with plusses and minuses. The simplest – and computationally fastest - methods are to either truncate or round to the nearest whole integer, with the penalty of producing a somewhat inferior image output. It is possible to perform first-order interpolation methods in order to compute in-between pixel values using the commonly used tri-linear interpolation algorithm described in [87]. Inverse affine mapping can also yield values that are outside of the original image boundaries, therefore it is important to catch these invalid indices and set the pixel value to some constant, normally black.

Figure 6-4 shows an original image and sample affine transformations with a few different parameter sets. This spatial remapping of images is a general technique that is used in the implementation sections of this thesis and will be demonstrated in several cases in Chapter 7. While affine transformations yield a spatial remapping, it is possible to create non-affine warpings of images as well. To generalize the re-mapping effect, let us return to Eq. 6-21 which merely alters the placement of pixels of one image to another. As long as we create a corresponding set of new spatial coordinates, this equation is still valid and can yield more entertaining results. Take, for example, the following inverse mapping equations:

$$\theta = \tan^{-1}\left(\frac{y' - y_c}{x' - x_c}\right) \tag{6-28}$$

$$d = \max\left(\sqrt{(x_c - s_w)^2 + (y_c - s_h)^2}, \sqrt{x_c^2 + y_c^2}, \sqrt{(x_c - s_w)^2 + y_c^2}, \sqrt{x_c^2 + (s_h - y_c^2)}\right) \tag{6-29}$$

$$r = \frac{\sqrt{(x' - x_c)^2 + (y' - y_c)^2}}{d} \tag{6-30}$$

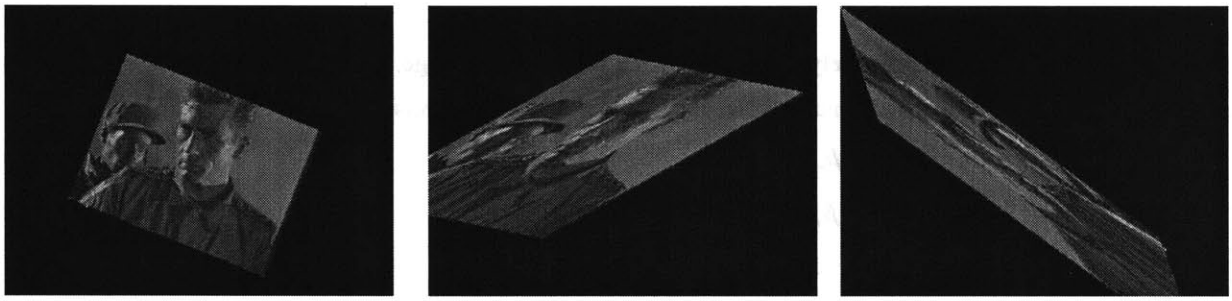

Figure 6-4. Three sample affine spatial transformations of a single frame.

$$x = r^{\gamma} d\cos(\theta) + x_c \tag{6-31}$$

$$y = r^{\gamma} d\sin(\theta) + y_c \tag{6-32}$$

where x_c and y_c are the coordinates of the center of the screen and γ is a "bow" parameter. s_w and s_h are the width and height of the screen, in pixels. The dependent variable, d, is important in order to properly scale the distances such that we keep within the bounds of the original image. This spatial remapping, using polar coordinates, will create a "fun-house" type mirror which will stretch out an image in proportion to the parameter γ.

The important issue to note with these two examples is the fact that there are a certain number of parameters that change the appearance of the visual reconstruction. With Eqs. 6-31 and 6-32 we have a user-controllable "bow" constant γ and an effect position x_c and y_c. Thus we can say that the reconstruction model has a three-dimensional parameter vector. Therefore, we need to create a mapping from the interface sub-system of the interactive system into this three-dimensional space. If we take our previously described Fish hand tracker interface that yields a three dimensional salient vector, (h_x, h_y, h_d), we need to provide a mapping function such that we move from one space to another, or $f:\Re^3 \rightarrow \Re^3$. With this reconstruction vector, we could try a simple mapping between the salient vector and the reconstruction vector:

$$x_c = h_c \quad y_c = h_y \quad \gamma = 1 + \frac{h_d}{D} \tag{6-33}$$

where D is a normalizing constant to scale the range of the bow. Using these parameter mapping functions will cause the interactive environment to produce a fun-house mirror whose center is placed where the user places their hand and whose severity of warp is inversely proportional to the distance that the hand is from the tracking camera. Although this is a purely subjective evaluation, it could be claimed that the spatial consistency between the viewer's hand and the center of the effect response makes for a more intuitive mode of interaction. A few sample outputs of this parameter mapping is shown in Figure 6-5.

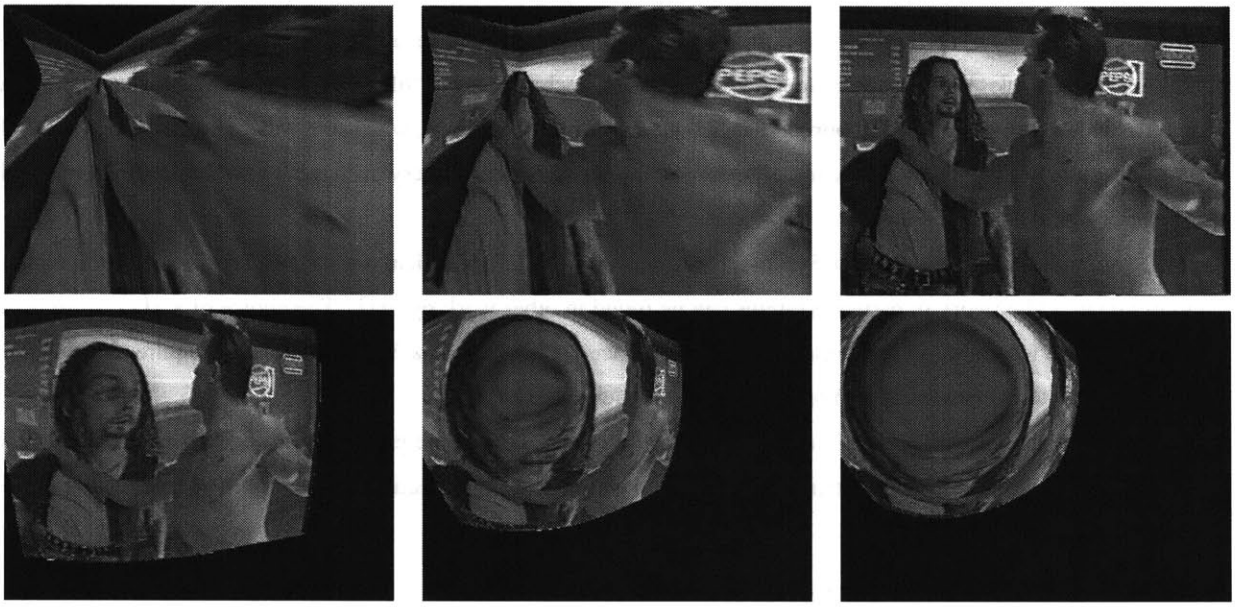

Figure 6-5. Six sample "fun-house" warping images with different "bow" parameters. Gamma values are (from top left to bottom right): 0.3, 0.6, 1.0 (normal image), 1.4, 2.0, 6.0.

6.4 Physical Based Simulations

The next type of reconstruction models is a more general hybrid that combines computer graphics with physical based modeling in order to produce an output that behaves, in a loose sense, much like systems we see in the real world. For example, researchers have used models of speech synthesis based on the analysis of the vocal track in order to produce realistic computer generated speech.[88] The keyword for physical based models is *simulation*, as the output should allude to that which it models. Ray tracing, a discipline of computer graphics, is a physical based model on how optics behave, simulating the effects of several known physical properties of light and motion, such as refraction through glass, specular reflections, and motion blur.[89]

In use with interactive installations, physical based simulations are very evocative as they help communicate an exposition of process. Keeping with the original goals that were outlined in Chapter 1, simulations are able to construct a meta-narrative without having to explicitly encode all of the potential states that the system can arrive at. As shown in the following discussion, these physical based models typically use very simple localize rules to describe the behavior of several independent operative agents. Although the rules of the agents are slight, very complex global patterns and qualities can arise. This is what is described as an emergent behavior where an objective, outside observer is able to organize all of the separate agents into a cohesive whole. Mitch Resnick uses very simple local rules to drive independent agents that simulate the foraging behavior of ant colonies.[90] In this case, it would be far more difficult to write a program from the "top-down" that would visualize this end effect as a centralized set of rules does not actually model the original itself.

6.4.1 Particle Systems

A particle system is based on simple Newtonian physics where a set of independent agents, particles, are acted upon by a group of forces, such as applied forces, from an outside source, and friction. These particles are set into a virtual environment over a time period and their motions indicate set resulting spatio-temporal forces that these highly localized rules describe. It is important to note that each particle is "blind" to the global environment, as they operate within a very small neighborhood at any give point in time. Thus, when one watches a particle system, the resulting global behavior of thousands of independent particles can be very surprising and not easily predictable, except for in the most simple cases.

The notion of particles in this sub-section is limited to a 2D graphical objects that are placed within a planar environment. Other work in particle systems can be found in other work by [91]. The choice of a 2D system is due to the compute feasibility in real-time on reasonably priced computers as a 3D system would require additional 3D (virtual space) to 2D (screen space) projections and rendering times.

Let us begin with a set of particles that have a current coordinate placement in the two dimensional space and a motion vector. As an initialization stage, we will assign a starting location for each of the particles. Let us then denote each particle as:

$$\mathbf{p} = (p_x, p_y, p_{\Delta x}, p_{\Delta y}) \tag{6-34}$$

but let us, to simplify the notation, break up the set of particles, numbered i=1....P, into four different vectors:

$$\mathbf{x} = \mathbf{p}_x \qquad \mathbf{y} = \mathbf{p}_y \tag{6-35}$$

$$\dot{\mathbf{x}} = \mathbf{p}_{\Delta x} \qquad \dot{\mathbf{y}} = \mathbf{p}_{\Delta y} \tag{6-36}$$

We will also group together the location and velocity components of a particle into two vectors:

$$\mathbf{l} = (\mathbf{x}, \mathbf{y}) \tag{6-37}$$

$$\mathbf{v} = (\dot{\mathbf{x}}, \dot{\mathbf{y}}) \tag{6-38}$$

For every output frame, at time t, we need to calculate the current position of each of the particles, which is simply:

$$\mathbf{l}_i[t] = \mathbf{l}_i[t-1] + \mathbf{v}_i[t-1] \quad \text{i} = 1....\text{P} \tag{6-39}$$

If there are no outside or friction forces in our particle system, then the particles will travel in its current initial direction. However this does not produce any particularly interesting results, so let us define a two dimensional force matrix:

$$\mathbf{F} = \begin{bmatrix} \mathbf{f}_{0,0} & \mathbf{f}_{0,1} & \mathbf{f}_{0,2} & .... & \mathbf{f}_{0,M-1} \\ \mathbf{f}_{1,0} & \mathbf{f}_{1,1} & \mathbf{f}_{1,2} & .... & \mathbf{f}_{1,M-1} \\ \mathbf{f}_{2,0} & \mathbf{f}_{2,1} & \mathbf{f}_{2,2} & .... & \mathbf{f}_{2,M-1} \\ : & : & : & & : \\ \mathbf{f}_{N-1,0} & \mathbf{f}_{N-1,1} & \mathbf{f}_{N-1,2} & .... & \mathbf{f}_{N-1,M-1} \end{bmatrix} \tag{6-40}$$

where each element of the force matrix is a two-dimensional force vector that "exists" at the x, y spatial coordinates that correspond with its indices:

$$\mathbf{f}[n,m] = \left(f_x[n,m], f_y[n,m]\right) \tag{6-41}$$

Here $f_x[i,j]$ and $f_y[i,j]$ are the force components in the x and y axes that are at coordinates (m,n) in the particle simulation plane. In order to refer to the force at any specific particle, i, at time t, we will use the following notation:

$$\mathbf{f}[x_i, y_i, t] = \mathbf{F}_{x_i, y_i}[t] \tag{6-42}$$

With this we can also update the velocity of each particle at each time step t:

$$\mathbf{v}_i[t] = \alpha\ \mathbf{v}_i[t-1] + \mathbf{f}[x_i, y_i, t-1] \quad \text{i} = 1....\text{P} \tag{6-43}$$

The constants α represents dampening force, or friction, against the current velocity. If the new forces drop to zero, then the velocity will also, when $\alpha < 1$, eventually decay to zero. Obviously, α should be set between 0.0 and 1.0 in order to insure stability of the system. Values greater than 1.0 will cause the particles to continually speed up exponentially without the presence of any external forces.

Several implementations of particle systems in interactive environments can be found in chapters 7 and 8 as part of Tod Machover's *The Brain Opera*.

Chapter

7 Implementation

7.1 From Theory to Practice

This chapter investigates several implementations of this model of interactivity. Although these test cases are limited in scope, the intent was to place as many of the theory principles into practice, as it is always through the engineering and realization that the full complexity of the problem reveals itself.

To me, one appealing aspect of interactive media art is that it forces the artist to be both a theorist as well as an engineer. While ideas may be provocative and stimulating, it is only through the construction of the interactive environment that a communication channel to a wider audience is formed. Typically avant-garde art has been reserved for the intellectual elite; however, now it is possible, with the use of the multimedia capabilities of computers, to enjoy an interactive work at multiple levels. Should an audience member not be well versed in the theories of Baudrillard and Benjamin, he/she can nevertheless enjoy the work at the visual and aural surface level. Conversely, those well versed in multimedia technologies and, therefore, not awed by its shimmering surface, can look for the artistic themes and meanings within the environment.

How the artist/engineer decides to implement the theory is a critical step, as it becomes the point at which all conceptual ideas must be given a mathematical body within the computational system. Should the implementation not fulfill the promises of the theory and thematic background, the work, in my opinion, has failed to become anything but hypothetical discourse. To me, as an artist and computer programmer, care should be taken to limit the scope of the work in order to make for a more direct implementation style. Therefore, my own pieces tend to be small well-focused experiments. *Winds that Wash the Seas* (1995) investigates the use of alternative interfaces within a simple cause-effect relationship. *What Will Remain of These?* (1997) focuses on the use of a physical based model coupled with computer vision optical flow analysis. However inviting the applications of computers in interactive art may be, it is the responsibility of the artist to make sure that there is a central theme that is appropriately implemented and presented to the audience, in my opinion. Otherwise, the work has a potential to become an unfocused mess that is held together by a hollow technological shell.

The thought process must be broken down into small components that can be programmed, tested, debugged, and executed, just like in any other software engineering project. Furthermore, depending on the scale of the project as well as the engineering skills of the artist, additional outside programming help must be hired and managed. Interactive media art production has thus been often related to film and video production, where it is difficult for a single person to implement all of the components necessary for a piece. Personally, my works are designed and implemented in whole by myself, with the exception of *The Brain Opera*, as described below. While this makes for consistency between the concept and the implementation, it also sometimes disallows the possibilities of receiving objective feedback during the implementation stages. One videotape work of mine, *Until We Sleep* (1995), uses many of the same abstraction/reconstruction strategies of this thesis. As it was not a real-time

interactive environment, the cause-effect mappings were very abstract and mathematically involved. While this intrigued myself, it was however not apparent to the audience what the underlying system was. Had I received critical feedback from someone, say a collaborator, a much tighter and more focused work would have been produced.

7.2 Design to Implementation Decisions

The process of going from design to implementation seems to be as related to compromise as it does in engineering. It is rarely the case that an implementation is exactly as one imagined during the conceptual design phase, since a number of typical issues, most notably cost and time constraints, arise during the production phase.

A good interactive artist, learning through multiple experiences, will keep these issues in mind during the design phase of a project. It is through the expectation of problems that designs can be made robust, easing the production and debugging phases. For example, one critical design criteria is, how long will this work be shown at a site? This all depends on the exhibition venue, and the artist should have different design concepts and implementation strategies in mind for different venues. Many times works are shown for relatively short times at festivals, from a few days to a few weeks. This means that the work will have several opportunities to be repaired and patched up after individual shows. But if an interactive environment is to be a permanent installation somewhere, then the design and engineering must be extremely robust and be able to withstand large volumes of people using it for long periods of time.

Furthermore, permanent or long-term installations should allow for multiple engagements so that when the audience returns to the exhibition, either the work has literally evolved over time or has multiple levels of engagement. The first contact with a piece could yield an aesthetical impression, while subsequent visitations could produce a more substantial interpretation, in terms of thematic content.

7.2.1 Design Concept

The design concept, or scenario, is largely a description of an "end-user" experience. As interactive media art uses the notion of experience as a central vehicle for artistic expression, it becomes a fundamental question to ask what type of experience a viewer/participant should receive. Should the experience, with the thematic content of the work in mind, be light or heavy, playful or profound? Should the environment reveal itself immediately, with all of the thematic content available at first sight, or should the user gradually reveal the content base? Although these adjectives would appear to sound very vague, they can form a starting point on which to base the production aspect of the project.

7.2.2 Length of experience

Another design question comes from the targeted length of experience a visitor will receive while engaged with an interactive work. Is the artist looking to captivate an audience for three to five minutes, or ten to 30 minutes? The length of engagement is also intimately linked to the exhibition venue; i.e. smaller galleries that draw a smaller

number of people allow for a more lengthy engagement of the work. For example, William Forsythe's work *Improvisational Technologies* was shown at the ZKM's MultiMediale 4 exhibition.[92] This intricate and subtle piece was for a single person who sat at a station and explored the styles of William Forythe's choreography as well as several examples of dance works. As the work drew in a single person for extended periods of time, lines formed of people waiting to experience the work. Many of the prospective visitors left the line frustrated after waiting too long. Furthermore, it was difficult for the user to lose themselves in the work as one was always conscious that a line of people was waiting to try it, making some of the participants feel rushed and uncomfortable. In my opinion, a smaller exhibition venue would have given the viewer a longer, more casual opportunity to dive into this engaging work without the pressures.

The *Interactive Mind Forest* section of Tod Machover's *Brain Opera*, described in Section 7.4 in this chapter, assumed a three to five minute engagement per person at each station. As the original concept of the *Brain Opera* was such that the audience would use the interactive lobby 30 to 45 minutes prior to the performance, it was important that each person would receive an opportunity to use each of the technologies. Given that each audience was estimated to be 100 to 200 people, the audience flow was an important issue to consider so that frustration levels and waiting times could be kept at a minimum. In Linz, Austria, during the Ars Electronica exhibition of the *Brain Opera*, a different setup was adopted where the *Interactive Mind Forest* was open to the public for approximately a week, with one night of two performances. This allowed people to come and go as they pleased and to spend as much time with each installation as they wished. Such an organization made for a more relaxed experience as it was rare that more than a few dozen people were using the *Mind Forest* at the same time.

7.3 Real-time Programming Issues

As interactive installations must adapt in response to the viewer's intent, one of the challenges of engineering such artworks lies in the development of very high speed systems that can produce an adequate experience in real-time. Real-time is a vague term in this case, as all interactive works are processed in real-time, but some much more than others. Works that reference a database of video content, such as hyper-video on CD-ROM, only need to read the video frames from secondary storage and place them on the screen, which is the least demanding for a computer from a system's point of view. Works that show computer generated imagery in response to the viewer need to be able to *render* the set of polygons and texture maps at a reasonable frame rate. Generally speaking, the artist should try to keep at least 12 to 15 frames-per-second; in order to avoid that the visual response is too jarring to the user, in my opinion. In particular, VR applications, where the viewer's entire visual field is completely computer generated, need a good frame rate to avoid unpleasantness. Furthermore, it is important to provide efficient real-time programming in order to insure a minimal time-lag between the actions of the viewer and the response of the system.

Again, the implementation questions are directly related to the choice of computation software tools that are used. There are several high-speed 3D rending packages such as Inventor and OpenGL by Silicon Graphics, that obviate many of the follow discussions.

Most important is the isolation of the compute critical portions of the software environment, or those functions and operations which take the most compute cycles to complete. Generally speaking, software programs that are used in interactive media have a few key components that take up most of the time. In many of my interactive works in the past, the overwhelming portion (~90%) of the compute time came from a small portion (~5%) of the program code. If the programmer can identify and optimize this critical section of code, the benefits will grow rapidly. Take for example this C code for alpha-blending two NTSC video frame buffers:

```
void alpha_blend(in1,in2,out,alpha)
PIXEL *in1,*in2,*out;
unsigned char alpha;
{
      int i,j,k;

      for(k=0;k<3;k++)
            for(i=0;i<640;i++)
                  for(j=0;j<480;j++)
                      *(out++) = ((*(in1++) * (256-alpha))+
                                  (*(in2++) * alpha)) >> 8;
}
```

Although this code fragment is, to the eye, very short, it can execute quite slowly on ordinary computers. In total we must blend 921600 pixels for a single NTSC image. This entails 1843200 multiplies, 921600 additions, 921600 bit shifts, 2764800 memory operations. Altogether, this comes to 6451100 operations, without considering the overhead of the for loops which would add on about another 2764800 operations, which would be expensive operations due to the execution branching that occurs in for loops. So what looks like a small function actually takes at least 10 MIPS away from the computer in five lines of program code. Considering that we need to compute this on a frame-by-frame level, there is clearly an upper bound to the speed of the system. The above full frame NTSC video blend code runs at about 17.6 frames per second on a 200MHz Pentium Pro computer.

On the other side of the coin, the non-critical - or control portions - of the software program generally can take up many more program lines. A software developer's tool, commonly known as a *profiler*, helps to determine how much percentage of total compute time has been distributed to various functions in a program.

Another important programming concern is whether to perform the computations in integer or floating-point mathematics. Almost all computers can do faster calculations in integer math and in many cases the difference between integer and floating-point calculations is substantial. For example the above alpha-blending code sample runs 8 times faster using integer math in comparison to floating point calculations on a 200MHz Pentium Pro!

7.4 Brain Opera

The Brain Opera, developed between 1995 and 1996 by Tod Machover and a group of 50 researchers at the M.I.T. Media Lab, is the first interactive opera that has been produced and presented to the wide public, premiering July 1996 at Lincoln Center, New York City. The work is based on Marvin Minsky's seminal book, The Society of Mind[93], in which he develops the idea of a decentralized set of agents that collectively form a recognizable intelligence, although, they themselves are dumb. Machover interpreted Minsky's ideas as being a metaphor for the

distribution of several simple musical devices that form rich and complex musical experiences. As the audience members moved from station to station, they received a blending of all of the distributed musical agents through the acoustical space.

Out of this overarching design, *The Brain Opera* consists of three major components:[94]

- **The Mind Forest**, a large installation area with 6 different interactive experiences: *The Speaking Tree*, *The Singing Tree*, *The Rhythm Trees*, *Harmonic Driving*, *Melody Easels*, and the *Gesture Walls*.
- **Net Music**, a set of World Wide Web Java applet pages in which remote users could both navigate musical content as well as participating in the live performance.
- **The Performance**, a live show with three performers using the same technology as in part of *The Mind Forest*. In addition, contributions from the spoken replies at *The Speaking Tree* portion of *The Mind Forest* as well as from the Internet Java instruments, were mixed in under the control of the performers.

The following sections will focus on the *Melody Easels* and *Gesture Wall* components of *The Mind Forest*. A photograph of the *Interactive Mind Forest* is shown in Figure 7-1.

7.5 Melody Easels

One of the interactive installations in *The Mind Forest* is a set of three *Melody Easels*, one of which is shown in Figure 7-2. The artistic concept of the *Melody Easels* was to create an active musical surface on which the participants could play various melody lines, controlling the phrasing of the piece through the smooth motions of their fingers. The interactive environment is much like a musical finger paint, smooth, continuous, as well as light and playful. Due to the commercial technology available at the time of development, each of the three *Melody Easels* is for a single active participant, although several additional spectators are accommodated for. The music consists of both an instrumental portion, generated in real-time through musical synthesizers, and a sampled

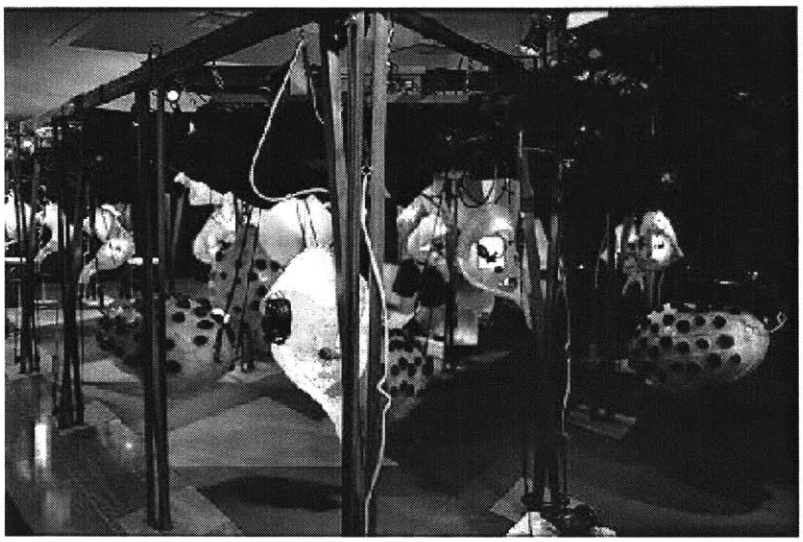

Figure 7-1. *Interactive Mind Forest* component of *The Brain Opera* as setup in Lincoln Center, NYC, July 1996. Shown in the foreground are *The Singing Trees*, *The Speaking Trees*, and *The Rhythm Trees*.

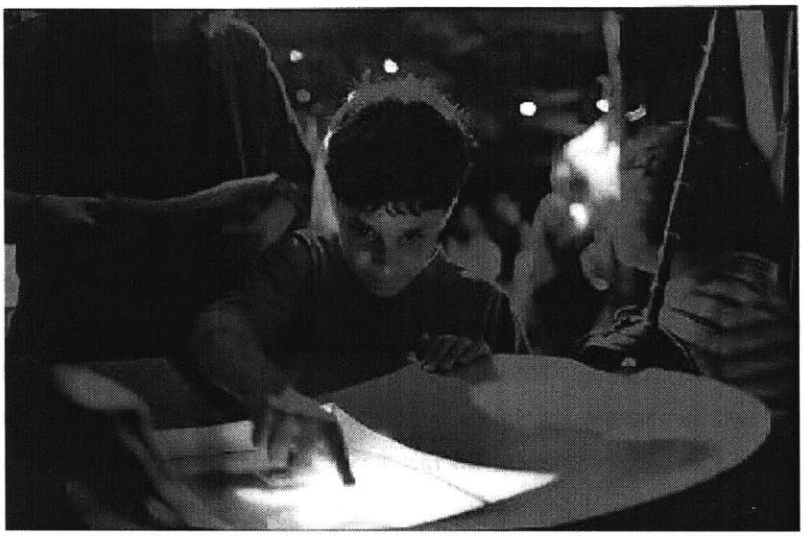

Figure 7-2. A participant enjoying one of the *Melody Easels* in *The Mind Forest.*

vocal melody line, sung by mezzo-soprano Lorraine Hunt. The concept and design of the *Melody Easel* was completed by Maggie Orth, Sharon Daniel, and Ray Kinoshita. The videographer Sharon Daniel developed the image content design which was decided to be related to murky, watery imagery that gave the Easel surface a very fluid contrast to the highly geometric structural design. The interactive musical system was designed by Kai-Yuh Hsiao who used the regularity of the finger motions in order to control music parameters of the musical composition written by Tod Machover.

The three *Melody Easels* are made from computer monitors that are suspended facing upwards by a set of three wires that are rung from the overhead set triangle. Each monitor is housed within a rounded, nose-cone shaped covering, with a solid bottom to protect the delicate monitor. Around the monitor glass is a slightly sloping aluminum covering that hides the less-than-attractive beige computer case from the viewer. In addition, the reflective qualities of the aluminum subtly diffused the imagery through the environment around the Easels.

7.5.1 Technology

The Melody Easels were developed with standard off-the-shelf technologies that included personal computers, touch screens, MIDI sound cards, and commercial sound synthesizers and samplers. This was done in part in order to keep maintenance at a minimum, as it would be relatively straight-forward to replace malfunctioning equipment as necessary. The Brain Opera was very fortunate to have generous sponsors who donated much of the equipment that was used in these installations. Table 7-1 is an exact list of equipment used for the development and exhibition of this installation. Figure 7-3 is a block-diagram outline of the entire system.

Each computer monitor uses an ELO touchscreen which offers three degrees of freedom: x, y, and pressure. Both the interactive visual and sound software systems use all of the information that is provided. Unfortunately, as mentioned above, the touch screen is only for a single user which broke the original intention of the installation. In

Company	Model Number	Equipment
IBM	PC750	Computer, 133MHz Pentium, 128 Mb RAM
ELO	IntelliTouch	Pressure-sensitive touchscreen
Akai	Prophecy	Synthesizer
Kurzweil	K2500	Rack mount sampler, 64 Mb RAM, Fujitsu hard drive
Mackie	1202	12 channel audio mixer
Samson	Servo 150	Studio amplifier, 75W stereo
KRK		100W speakers (2)

Table 7-1. *Melody Easel* equipment list.

several situations, more than one person attempted to play the *Melody Easel*, causing errors stemming at the touchscreen hardware level. The exhibition computers were 133MHz Pentium computers, which, at the time of writing, represent rather the lower end of the personal computer spectrum. Furthermore, no graphic accelerator cards were used at all in the system, which created an additional implementation constraint in terms of bitmap rendering speeds.

7.5.2 Interactive Visual Software Implementation

The following sections explain the development of the interactive computer graphics portion of each the three *Melody Easels*. The software was written by myself, under the conceptual direction of Sharon Daniel. As mentioned before, the visual software was running on the same computer as the music generation software, so there were many tasks running concurrently, creating an even tighter allocation of compute resources.

We developed a unique visual effect for each of the Easels, although there are several similarities between them. Sharon Daniel digitized approximately a dozen short video sequences that used water in come capacity. For example, several shots were completed in a pool or a pond with an actress making floating gestures in the water. When the interactive viewer looked into the Easel, it evoked a feeling of looking into a kettle, as the design of the

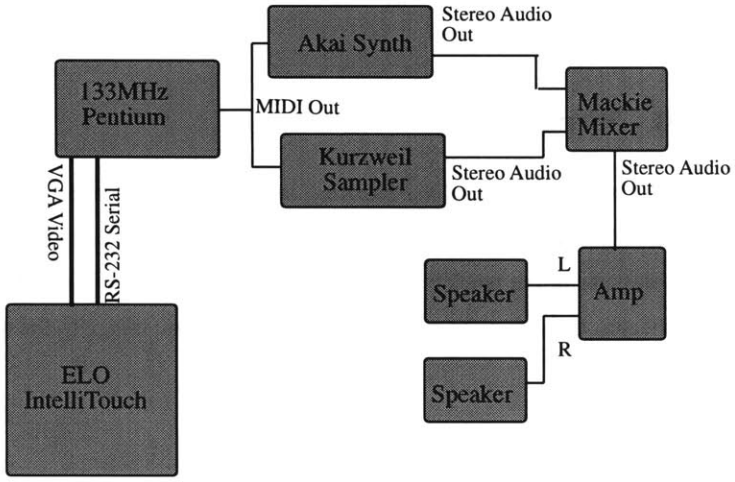

Figure 7-3. System block diagram of the Melody Easel.

structure made the objects themselves look like cauldrons. These sequences were short, around ten seconds each, and were looped continuously. In order to avoid that the installation becomes too repetitious, each *Melody Easel* uses several video layers, or video sequences which appear one behind each other, so that viewers can navigate through their interaction. The software concept had the intention - should the participant engage vigorously enough in the environment - they should transition from layer to layer, progressing further down the sequences of the video. This was to be a motivational technique, encouraging the viewer to further explore the environment and the different layers of imagery that it contains.

The video sequences were stored as color bitmaps with 16-bpp (bits-per-pixel) RGB information. As bitmapped video is very large in terms of memory requirements, and slow to read from secondary storage, the entire image sequence was placed in a large bank of DRAM. We chose to use bitmaps in order to facilitate image processing routines, as it is not possible to easily transform compressed images. Typically, one would have to decompress the image first, apply the transformation and then render that image to the screen. This decompression stage would have further required already strained compute resources, and therefore the reduction of the data size did not justify the additional increase in compute time. This is one of the "catch-22's" of using data compression: one can reduce the image representation size, proportionally reducing the reading time from a hard disk, but then at the cost of not being able to perform operations on the image.

Each of the *Melody Easels* used image processing techniques in order to create transformations and abstractions of these pre-recorded sequences. This was chosen in order to create a playful and suggestive environment where each *Melody Easel* can exhibit an interactive behavior that subtly modifies the video sequences. Although overall performance may have been increased through the use of 2D computer graphics animation, it is difficult to keep the imagery soft and abstract. 2D computer graphics tend to prefer rigid and geometric forms, a style of figurative expression that we chose to avoid.

Due to computational constraints, the interactive visual environment used imagery at a resolution of 320 x 240 pixels that were interpolated to 640 x 480 (NTSC size) at the rendering stage. To perform the calculations at the native 640 x 480 would have incurred too great a cost – 4x slower – without considering additional problems with the level 2 cache.

7.5.3 Melody Easel #1 – Blend

The first *Melody Easel* uses a spatio-temporal alpha-blending image processing algorithm with the effect, that the user can "rub away" a top layer of video, revealing the contents of another video sequence underneath.

The position and the pressure of the participants' finger on the touchscreen is measured and used to control a alpha blend matrix which indicates, on a pixel-by-pixel basis, the ratio of one image to the other. Let us represent the alpha blend matrix:

$$\mathbf{A} = \begin{bmatrix} \alpha_{0,0} & \alpha_{0,1} & \alpha_{0,2} & .... & \alpha_{0,M-1} \\ \alpha_{1,0} & \alpha_{1,1} & \alpha_{1,2} & .... & \alpha_{1,M-1} \\ \alpha_{2,0} & \alpha_{2,1} & \alpha_{2,2} & .... & \alpha_{2,M-1} \\ : & : & : & & : \\ \alpha_{N-1,0} & \alpha_{N-1,1} & \alpha_{N-1,2} & .... & \alpha_{N-1,M-1} \end{bmatrix} \tag{7-1}$$

where each element in this matrix is a real number valued between 0.0 and 1.0. The indexing of this matrix corresponds directly to the spatial coordinates of the output image. Thus element $\alpha_{i,j}$ is the alpha blending constant at screen coordinate (j,i). A value of 0.0 represents the full weighting of the top layer of the video at the specified coordinate and, conversely, a value of 1.0 will fully weight the background pixel.

To create an output mixing of the two video images, it merely has to perform a weighted averaging at each pixel:

$$\bar{\mathbf{I}}_{i,j} = (1.0 - \mathbf{A}_{i,j})\,\mathbf{I}^0_{i,j} + \mathbf{A}_{i,j}\,\mathbf{I}^1_{i,j} \qquad i = 0...N-1, j = 0...M-1 \tag{7-2}$$

As the *Melody Easels* have a temporal component as well, we decided that after the user rubs away a portion of the imagery, the blending will fade back to the top layer of video gradually over time. Therefore the alpha blending matrix should be a function of time as well:

$$\mathrm{A}[t] = \begin{bmatrix} \alpha_{0,0}[t] & \alpha_{0,1}[t] & \alpha_{0,2}[t] & .... & \alpha_{0,M-1}[t] \\ \alpha_{1,0}[t] & \alpha_{1,1}[t] & \alpha_{1,2}[t] & .... & \alpha_{1,M-1}[t] \\ \alpha_{2,0}[t] & \alpha_{2,1}[t] & \alpha_{2,2}[t] & .... & \alpha_{2,M-1}[t] \\ : & : & : & & : \\ \alpha_{N-1,0}[t] & \alpha_{N-1,1}[t] & \alpha_{N-1,2}[t] & .... & \alpha_{N-1,M-1}[t] \end{bmatrix} \tag{7-3}$$

In order to accommodate the feature of the gradually fading back to the foreground image, all we need to do, at each time step, is to multiply the weighting matrix by a scalar constant valued between 0.0 and 1.0:

$$\mathbf{A}[t] = \beta\,\mathbf{A}[t-1] \tag{7-4}$$

This works because of the way we assigned the mappings of the alpha blend elements, where 0.0 is full foreground. Had we not chosen this appropriate representation, then this desired artistic operation of "fading back" would have been more difficult than a simple multiplication. The apparent speed at which this matrix returns to zero is related to the value of β. If β is very close to 1.0, then the decay will be slow. Conversely, a somewhat small value of β will cause the alpha blending matrix to drop very rapidly. The reader should keep in mind that at each time step t, we are multiplying the entire matrix by this value. Therefore the decay of the matrix is exponential in time. If we have a value of β =0.93 and t=10, then we have the flowing relationship:

$$\mathbf{A}[10] = 0.93^{10}\,\mathbf{A}[0] = 0.484\mathbf{A}[0] \tag{7-5}$$

As you can see this drops off very quickly, in only ten time steps, we have cut our original blending matrix by more than half! Keeping in mind that we need to divide this figure by the number of frames per second throughput, ten time steps represents a very short portion of time, i.e. about 0.5 seconds at 20 frames per second. Figure 7-4 shows

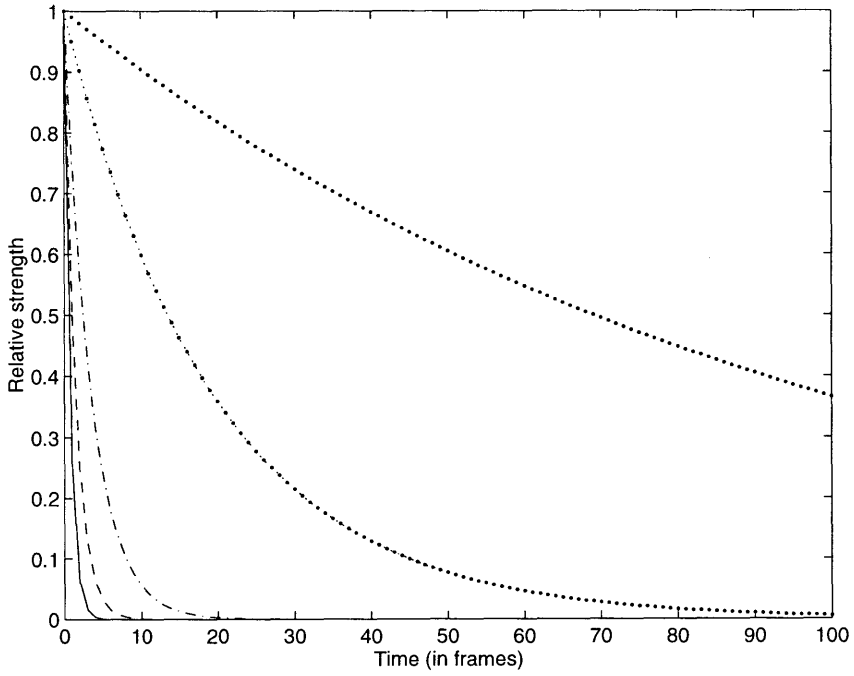

Figure 7-4. Decay rates for five values of β: 0.25, 0.5, 0.75, 0.95, and 0.99.

several choices for β and their corresponding decay rates. In general, values between 0.95 and 1.0 are used in order to achieve a perceivable effect, otherwise the fade-away occurs too quickly.

Now that we have defined the behavior of the alpha matrix over time, let us discuss the interactivity between the user and the computational system. Let us define another matrix that represents the information from the touch screen:

$$\mathbf{S} = \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & \cdots & s_{0,M-1} \\ s_{1,0} & s_{1,1} & s_{1,2} & \cdots & s_{1,M-1} \\ s_{2,0} & s_{2,1} & s_{2,2} & \cdots & s_{2,M-1} \\ \vdots & \vdots & \vdots & & \vdots \\ s_{N-1,0} & s_{N-1,1} & s_{N-1,2} & \cdots & s_{N-1,M-1} \end{bmatrix} \tag{7-6}$$

where each element of the matrix is the touch information at each coordinate on the screen. Notice that this matrix is the same size as the N x M image matrices as well as the alpha mixing matrix. The ELO touch screen drivers produce mouse motion events under Microsoft Windows operating system, from which we get a x, y, and pressure information - here normalized between 0.0 and 1.0 and denoted as p. At each mouse motion event we assign a value to this matrix accordingly:

$$s_{i,j} = \begin{cases} p * \left(1.0 - \left(\dfrac{1}{\sqrt{(j-x)^2 + (i-y)^2}} \right)^{\lambda} \right) & \text{if } \sqrt{(j-x)^2 + (i-y)^2} < p * d_{\max} \\ 0 & \text{otherwise} \end{cases} \tag{7-7}$$

This equation basically assigns a value for the touch screen matrix that is proportional to the pressure that the viewer is applying to the screen and inversely proportional to the distance between the matrix element and the actual touch location. d_{max} is the maximum distance away from the actual touch coordinates that this assignment operates. This is used to help limit the number of calculations required in the software. Notice that this is scaled by the pressure measurement, creating larger and smaller video holes. The exponent constant λ is used to determine the slope of the spatial decay and should be typically greater than 1.0.

Once this touch screen matrix is defined we can simply combine Eq. 7-4 and Eq.7-7 into:

$$\mathbf{A}[t] = \text{clip}(\beta\, \mathbf{A}[t-1] + \mathbf{S}[t]) \tag{7-8}$$

Here both elements, the interactive component as well as the systemic behavior, are united into a relatively straightforward equation. The clip function limits the all of the elements in the matrix between 0.0 and 1.0. This is necessary in case multiple touch-screen events occur quickly near similar screen coordinate locations, as values above 1.0 in the alpha-blend matrix has an undefined meaning. Note that the interactive touch screen matrix $\mathbf{S}[t]$ is defined to be zero when there is no touch screen event, i.e. the finger pressure is 0.

To reconstruct the interactive system at every time step, all we need to do is use the current value of $\mathbf{A}[t]$:

$$\bar{\mathbf{I}}_{i,j}[t] = (1.0 - \mathbf{A}_{i,j}[t])\,\mathbf{I}^0{}_{i,j}[t] + \mathbf{A}_{i,j}[t]\,\mathbf{I}^1{}_{i,j}[t] \quad i = 0...N-1, j = 0...M-1 \tag{7-9}$$

Figure 7-5 shows several sample output images of the "Blend" version of the *Melody Easel*. Notice how the interactive viewer can "wipe away" spatial sections of the video image to reveal another level underneath. With the alpha blending decay, a "trail" will follow the participant's finger, gradually fading away. The effect is very soft and continuous, due to this slowing fading trail as well as the rounding of the input stimulus around the finger as detailed in Eq. 7-6.

Also, at each time step, we want to calculate an estimate of how much of the top layer of video is exposed. This can simply be done by an element summation of the alpha blending matrix:

$$e[t] = \sum_{i=0}^{N-1}\sum_{j=0}^{M-1} \alpha[t]_{i,j} \tag{7-10}$$

When this summation is greater than some pre-defined constant, say:

$$\mu = N * M * \phi \tag{7-11}$$

where ϕ is a percentage of the image "revealed", we can trigger an event. Thus when the viewer clears away enough of the top layer of the video, the system goes into a transition mode that fully brings the underlying video sequence to the front while loading in the next video sequence from the hard drive. At this point the old lower video level becomes the top layer and the video sequence that is has been loaded is the new bottom layer. The alpha matrix is reset to 0, showing only the top level.

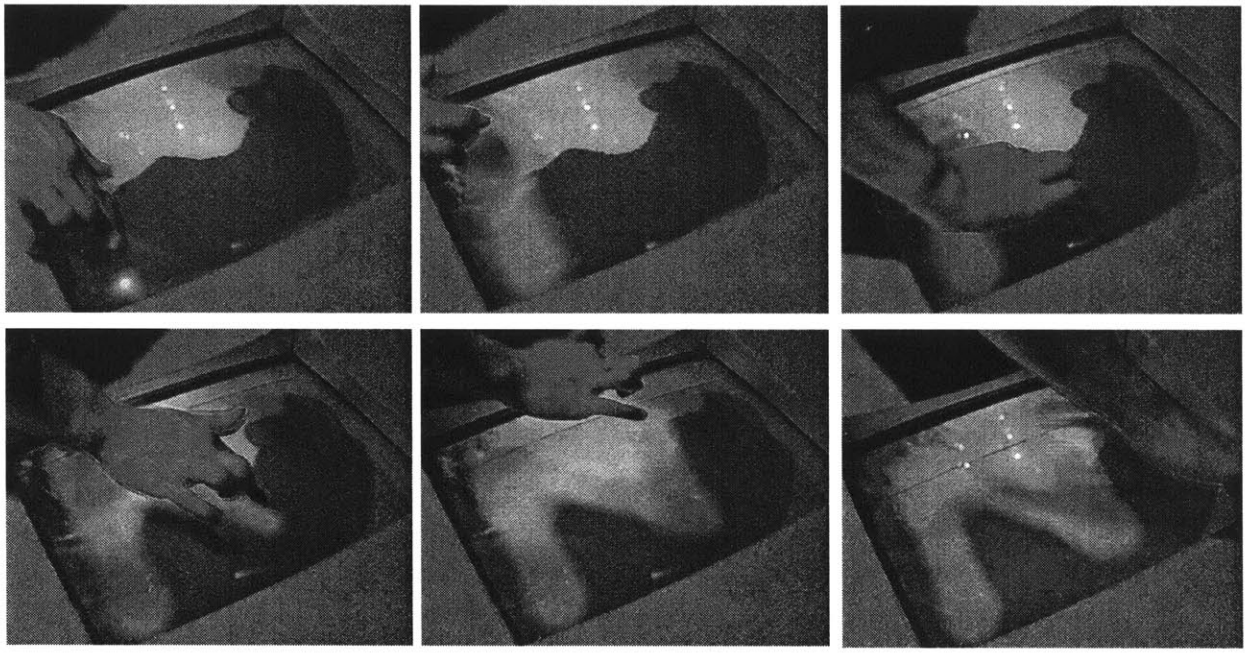

Figure 7-5. "Blend" *Melody Easel* in use. The participant, with his finger, "wipes" away the top video layer to reveal another sequence underneath.

So with this interactive system, the representation of the user is in three parameters that form a salient vector of:

$$\mathbf{u}[t] = (x[t]_{finger}\ ,\ y[t]_{finger}\ ,\ p[t]_{finger}\ ) \tag{7-12}$$

which, although very sparse, is what has been enabled through the interface technology, i.e. the touch screen.

7.5.4 Melody Easel #2 – Water Ripples

The second *Melody Easel* uses an interactive visual mapping in which the video screen behaves like a small body of water, reflecting the imagery of the video sequences. When the viewer moves his/her finger on the screen surface, water ripples originate from the place of contact on the screen and moves outwards towards the edges. These waves were not pre-rendered animations but rather used the spatial remapping techniques that are discussed in Chapter 6. As the viewer moves his/her finger across the touch screen, a rippling effect grows in accordance with the motions.

In order to achieve this effect, a physical-based model of a spring system was simulated within the interactive environment. In a spring system, energy is distributed along physical connections that perpetuate a force along the connections. In what is known as an over-damped spring system, the conveyed energy decays over time due to friction. Thus the sum of energy that is distributed over the connection is less than the energy that was received in. Conversely, under-damped spring systems constantly accumulate energy, oscillating wildly. Although almost all real world examples of spring systems are – thankfully - over-damped, a few architectural disasters have occurred when a construction actually is under-damped under certain conditions such as harmonic resonance.

Ideally it would have been most appropriate to have the spring system work at the pixel level, with virtual springs between each neighboring pixel. As one pixel is disturbed by an impulse energy, the motion would radiate outwards. However, due to the compute resources at hand, it was required to abstract the problem a little bit more. Rather than compute the spring system on a pixel-by-pixel level, a series of control points that form a virtual grid over the image was used. Each control point would be connected to its eight neighbors.

Let the matrix **X** represent all of the x components and matrix **Y** represent all of the *y* components of the spatial coordinates of the control points:

$$\mathbf{X} = \begin{bmatrix} x_{0,0} & x_{0,1} & x_{0,2} & \cdots & x_{0,M_c-1} \\ x_{1,0} & x_{1,1} & x_{1,2} & \cdots & x_{1,M_c-1} \\ x_{2,0} & x_{2,1} & x_{2,2} & \cdots & x_{2,M_c-1} \\ \vdots & \vdots & \vdots & & \vdots \\ x_{N_x-1,0} & x_{N_x-1,1} & x_{N_x-1,2} & \cdots & x_{N_x-1,M_c-1} \end{bmatrix} \tag{7-13}$$

$$\mathbf{Y} = \begin{bmatrix} y_{0,0} & y_{0,1} & y_{0,2} & \cdots & y_{0,M_c-1} \\ y_{1,0} & y_{1,1} & y_{1,2} & \cdots & y_{1,M_c-1} \\ y_{2,0} & y_{2,1} & y_{2,2} & \cdots & y_{2,M_c-1} \\ \vdots & \vdots & \vdots & & \vdots \\ y_{N_x-1,0} & y_{N_x-1,1} & y_{N_x-1,2} & \cdots & y_{N_x-1,M_c-1} \end{bmatrix} \tag{7-14}$$

where N_c and M_c are the number of control point rows and columns, respectively.

These two matrices are functions of time in this system and their behavior is calculated through the simple spring relationship:

$$\begin{aligned} x_{i,j}[t] = x_{i,j}[t-1] &+ \beta * \delta x_{i,j-1}[t-1] + \beta * \partial x_{i,j+1}[t-1] + \beta * \delta x_{i-1,j}[t-1] + \beta * \delta x_{i+1,j}[t-1] + \\ &\beta * \delta x_{i-1,j-1}[t-1] + \beta * \delta x_{i+1,j-1}[t-1] + \beta * \delta x_{i-1,j+1}[t-1] + \beta * \delta x_{i+1,j+1}[t-1] \end{aligned} \tag{7-15}$$

$$\begin{aligned} y_{i,j}[t] = y_{i,j}[t-1] &+ \beta * \delta y_{i,j-1}[t-1] + \beta * \partial y_{i,j+1}[t-1] + \beta * \delta y_{i-1,j}[t-1] + \beta * \delta y_{i+1,j}[t-1] + \\ &\beta * \delta y_{i-1,j-1}[t-1] + \beta * \delta y_{i+1,j-1}[t-1] + \beta * \delta y_{i-1,j+1}[t-1] + \beta * \delta y_{i+1,j+1}[t-1] \end{aligned} \tag{7-16}$$

where:

$$\delta x_{i,j}[t] = x_{i,j}[t] - x_{i,j}[0] \tag{7-17}$$

$$\delta y_{i,j}[t] = y_{i,j}[t] - y_{i,j}[t] \tag{7-18}$$

and $x_{i,j}[0]$ and $y_{i,j}[0]$ are the initial spatial positions of the control point within the matrix. The constant β is the forcing dampening coefficient. This equation couples the spatial relationships between neighboring control points over time. Of course, true physical systems would not have this time step delay, but in this case it will serve as a useful approximation. As one control point is displaced from its initial starting point, or a point of no potential energy, it "pushes" or "pulls" the other control points accordingly.

The control points are associated with a particular region in the source image. As the points deviate from their initial starting point, they need to stretch and distort the image correspondingly. We can use these control points to drive the parameters of an affine image transformation. All we must do is compute the affine transformation parameters. So we know the original initial coordinates of the control points and we know the current position. Thus, it is simple to compute the affine transformation matrix, as defined in Chapter 6. However, in this

version of the *Melody Easel*, the computer was not fast enough to perform the full transformation in real-time, due to many other software tasks executing concurrently. We used an approximate affine transformation which was a compromise between the accuracy of the full transformation and the speed efficiency of a basic tri-linear interpolator. If we perform this operation for each image region, we get a complete image warp that can have a very complex distortion.

This *Melody Easel* system becomes interactive through the touch of the viewer, which provides the initial control point displacement that brings the surface of the image to life. The touch screen interactive matrix is defined here to give the control points their initial displacement.

$$\mathbf{S}=\begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & \cdots & s_{0,M_c-1} \\ s_{1,0} & s_{1,1} & s_{1,2} & \cdots & s_{1,M_c-1} \\ s_{2,0} & s_{2,1} & s_{2,2} & \cdots & s_{2,M_c-1} \\ \vdots & \vdots & \vdots & & \vdots \\ s_{N_c-1,0} & s_{N_c-1,1} & s_{N_c-1,2} & \cdots & s_{Nc-1,M_c-1} \end{bmatrix} \tag{7-19}$$

where we define each element in this matrix accordingly, given a user representation of $\mathbf{u}=(x,y,p)$ that indicate the x and y coordinates of the touch event and the pressure estimate p:

$$s_{i,j}=\begin{cases} p*\gamma*\left(\dfrac{1}{\sqrt{(i-x)^2+(j-y)^2}}\right)^{\lambda} & \text{if } \sqrt{(i-x)^2+(j-y)^2} < p*d_{\max} \\ 0 & \text{otherwise} \end{cases} \tag{7-20}$$

where λ is the decay rate of the disturbance, as defined in Eq. 7-7 and γ is a scalar value to change the amplitude of the impulse.

However, we need to convert this matrix into a pair of displacement matrices, $\Delta\mathbf{X}$ and $\Delta\mathbf{Y}$, that describe the displacement of a control point in terms of x and y directions. This can be accomplished by the following equations:

$$\Delta x_{i,j}=s_{i,j}*\frac{j-x}{\sqrt{(j-x)^2+(i-y)^2}} \tag{7-21}$$

$$\Delta y_{i,j}=s_{i,j}*\frac{i-y}{\sqrt{(j-x)^2+(i-y)^2}} \tag{7-22}$$

which merely scale, by $s_{i,j}$, the unit vector between the user's finger and each control point. Control points far away from the disturbance point are not affected. Therefore in order to integrate this interactive matrix into the system, we need to, at each time step, add these displacement matrices to the positions:

$$\begin{aligned} x_{i,j}[t]=x_{i,j}[t-1]+\beta*\delta x_{i,j-1}[t-1]+\beta*\partial x_{i,j+1}[t-1]+\beta*\delta x_{i-1,j}[t-1]+\beta*\delta x_{i+1,j}[t-1]+ \\ \beta*\delta x_{i-1,j-1}[t-1]+\beta*\delta x_{i+1,j-1}[t-1]+\beta*\delta x_{i-1,j+1}[t-1]+\beta*\delta x_{i+1,j+1}[t-1]+\Delta x_{i,j}[t] \end{aligned} \tag{7-23}$$

$$\begin{aligned} y_{i,j}[t]=y_{i,j}[t-1]+\beta*\delta y_{i,j-1}[t-1]+\beta*\partial y_{i,j+1}[t-1]+\beta*\delta y_{i-1,j}[t-1]+\beta*\delta y_{i+1,j}[t-1]+ \\ \beta*\delta y_{i-1,j-1}[t-1]+\beta*\delta y_{i+1,j-1}[t-1]+\beta*\delta y_{i-1,j+1}[t-1]+\beta*\delta y_{i+1,j+1}[t-1]+\Delta y_{i,j}[t] \end{aligned} \tag{7-24}$$

This set of equations define both the behavioral and the interactive components of the "Water Ripple" *Melody Easel*. Figure 7-6 shows several examples of the user interacting with this version of the *Melody Easel*. Notice how a rippling distortion is formed on top of the video sequences as the user moves his/her finger.

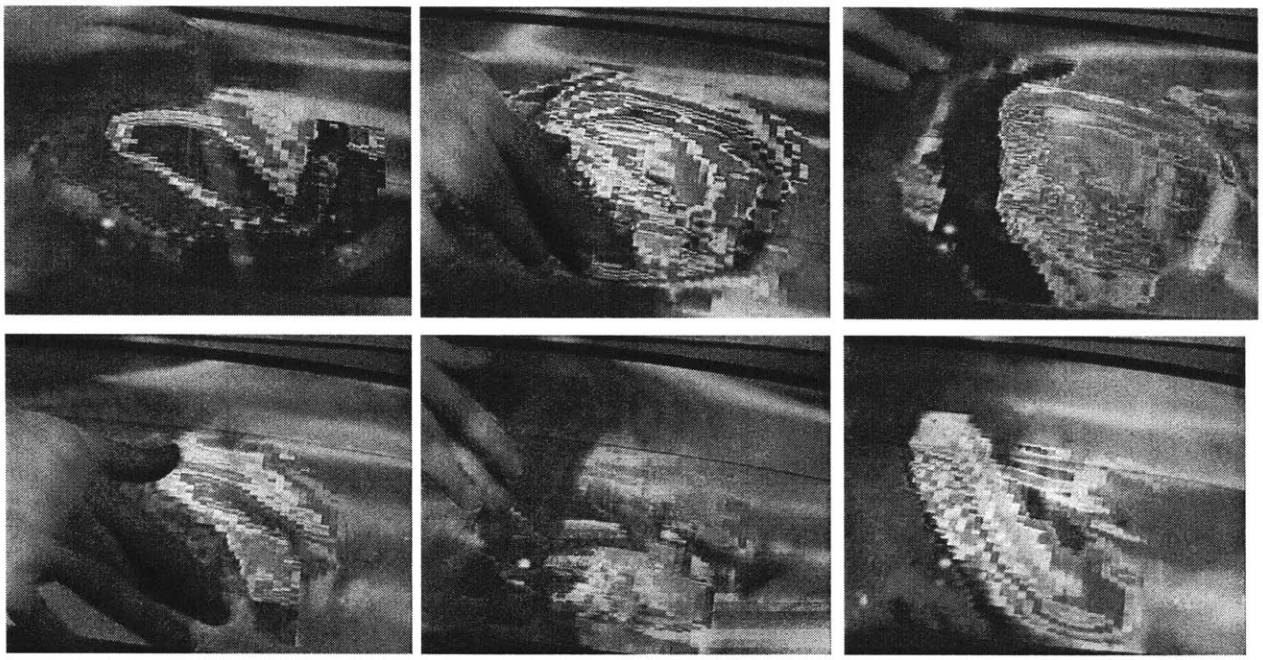

Figure 7-6. "Water Ripple" *Melody Easel* in operation. Wherever the participant presses, a water wave emanates from his fingertips.

7.5.5 Melody Easel #3 – Tin

The second *Melody Easel* behaves as if the touchscreen area was a piece of thin gauge tin that, when pressed, "bends" and "creases" the underlying video imagery. The viewer can deform the rigid video sequences into a distorted and abstract collage of shape, texture, and color. This effect came to life, ironically enough, through a programming error while I was developing *Melody Easel* #2, proving once again that good results can come from bad engineering. The underlying software is similar to *Melody Easel* #2 where there is a set of control points on an image plane. However the concept was for the image to be made like "silly putty" streching out and then returning back to its normal form automatically. Unlike the water ripple scenario, there was no need to create a connected spring system, simplifying the design somewhat.

However, we need to define a single spring force that will bring the control point back to its origin over time when no user force is applied on the touch screen. Using the *Melody Easel* #2 as a basis, let us modify Eqs. 7-23 and 7-24 to ignore the affects of the neighboring control points and add a force term for the "return to home" behavior:

$$x_{i,j}[t] = x_{i,j}[t-1] - \beta * \delta x_{i,j}[t-1] + \Delta x_{i,j}[t] \tag{7-25}$$

$$y_{i,j}[t] = y_{i,j}[t-1] - \beta * \delta y_{i,j}[t-1] + \Delta y_{i,j}[t] \tag{7-26}$$

Note the change of the sign and indices of the second term of both equations. The change of sign is due to the fact that we want the force to be directed *inwards*, towards the control point's origin. Again, as before, β is a spring constant that determines how fast a control point should return to its point of origin. During the exhibition, relatively

small values of β were used in order for the image to slowly return to normal. Through experimental aesthetical testing, we found that too large values of β did not yield a very interesting result as the surface was unwarping itself at a fast rate. When the image returned to normal too quickly, it was not possible to add more stretches on top of each other. Given the slow return rate, very complex and intricate distortions could be build up over time by multiple participants. Only when this *Melody Easel* was left alone for some time, would the image return completely back to normal.

The rest of the system design of the *Melody Easel* #3 is the same as #2; the control points drive an affine transformation that remaps the image according to how the surface has been distorted. Since this is described in detail in the previous sub-section, it will not be repeated here.

The transition criterion was a simple derivation of the state of the control points. The visual conceptual idea was that when a participant "crumpled" up the image enough so that it was far beyond being recognizable, the system would load in another video sequence secretively. When the control points gradually return to their origins, a completely new image would unfold in front of the viewer's eyes! This was the reward for the participant's hard effort to interact so vigorously with the environment. Figure 7-7 shows several images from the Melody Easel #3.

7.6 Gesture Walls

Another of the major *Mind Forest* components are the five *Gesture Walls* that are grouped together into one corner of the interactive lobby experience. As the name implies, the *Gesture Walls* are intended to provide an environment which is sensitive to physical gestures on the part of the participant. In comparison with other experiences in the *Mind Forest*, like the *Melody Easels*, there are no objects that serve as points of reference. This turned out to be both an advantage and a liability. Since there are no such tangible objects, the environment is relatively free from viewer associations as there is little that is presented to the participant except for empty space.

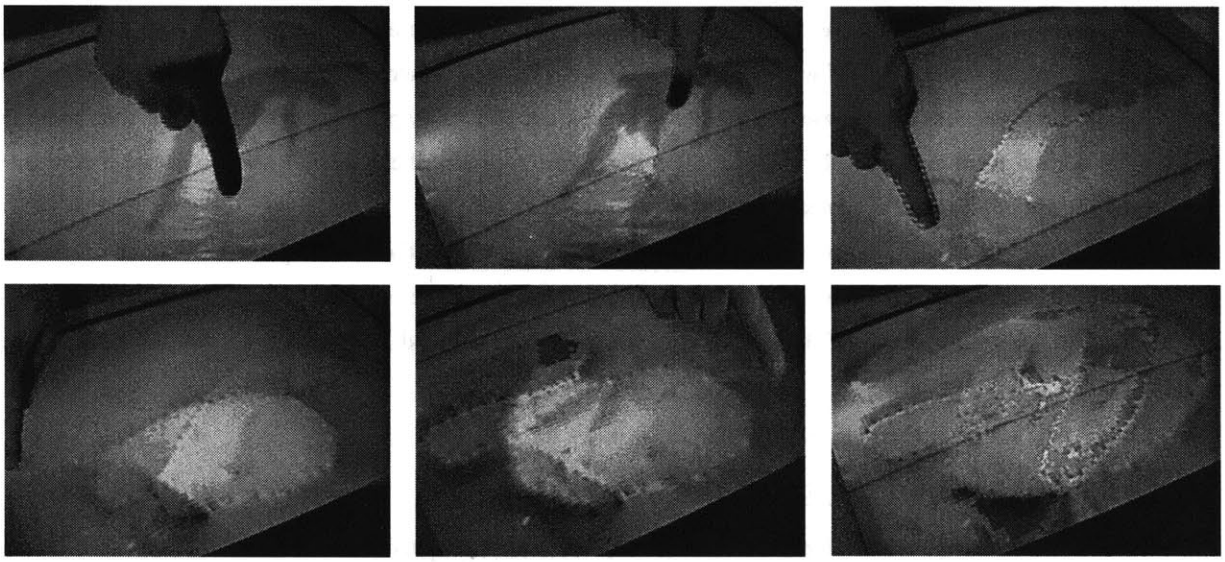

Figure 7-7. "Tin" *Melody Easel* in operation. The participant "bends" the image surface with his finger.

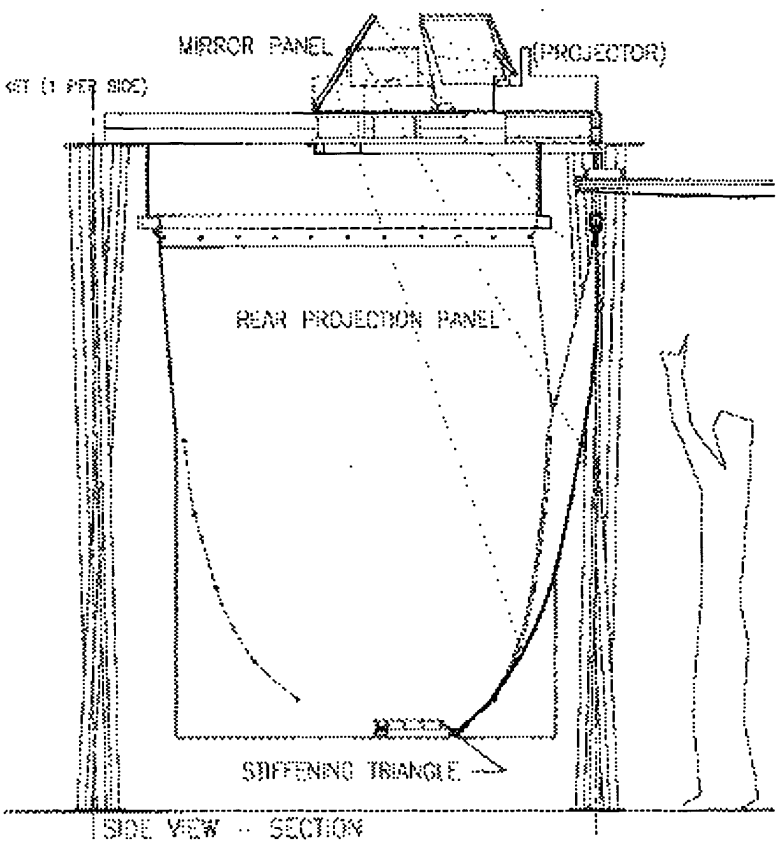

Figure 7-8. Mechanical drawing of the *Gesture Wall*.

Figure 7-8 shows a detailed mechanical drawing of a *Gesture Wall*. The artistic concept and design of the physical structure was created by Ray Kinoshita, Sharon Daniel, and Maggie Orth. The *Gesture Walls* were designed as an open space in which the viewer could freely move about and make physical gestures with his/her whole body. These abstract gestures, which are made in the free space, are mapped into sound and image transformations in accordance to the motions. The musical software was written by Kai-Yuh Hsiao that mapped the spatial information from the interface to amplitude, pitch, and instrumentation parameters of another musical work by Tod Machover. The videography was conducted by Sharon Daniel, keeping in line with the "water" theme of the *Melody Easels*.

Structurally speaking, the *Gesture Walls* are large, looming in front of the viewer. A translucent material is hung from the triangle frame, onto which the video imagery is back-projected. Speakers are also suspended in the air producing the musical feedback into the environment. On the floor directly in front of the screens, is a copper metal triangle on which the interactive viewer must stand. At the back end of this triangle is a "calibration" unit for the interface, which is described later. Last, four protruding "buds" at the ends of metal goosenecks are bolted to the frame structure, forming a rectangle between the viewer and the screen. These buds are the "Fish" interface devices that are described shortly hereafter in section 7.6.1.

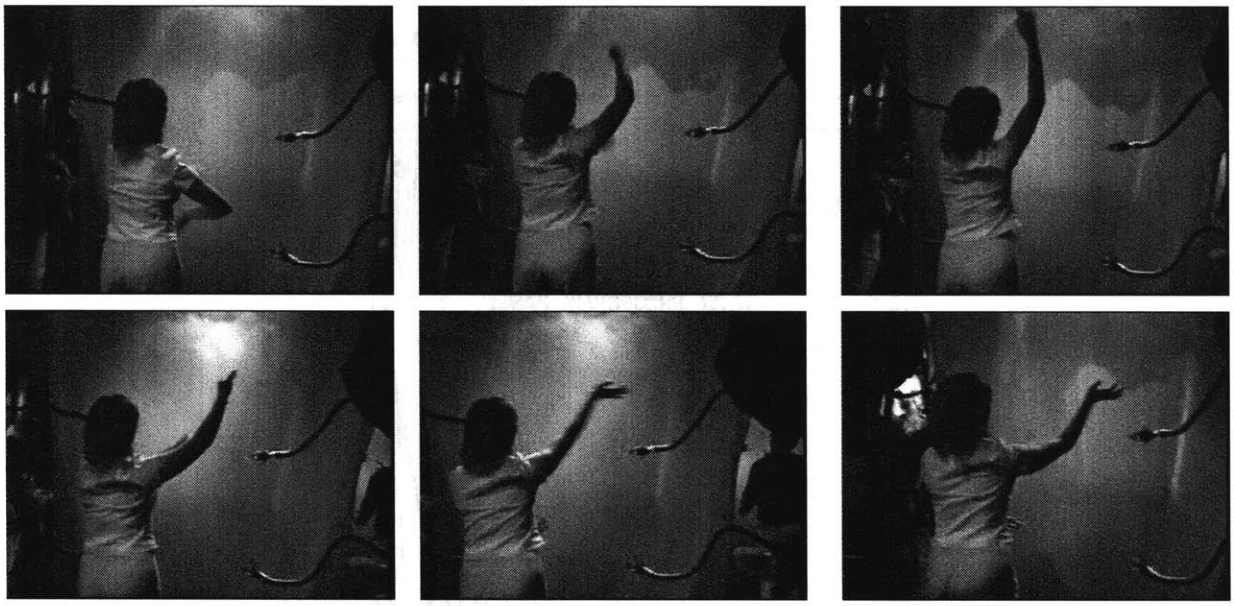

Figure 7-9. A woman interacting with one of the *Gesture Walls*.

Each one of the five *Gesture Walls* is independent from each other, allowing for five different participants to interact simultaneously. When standing away from the collection of the installations, the non-participatory audience members receive a rich blending of musical melody lines that overlap in the *Mind Forest*. One of the *Gesture Walls* is shown being used in Figure 7-9.

7.6.1 Technology

The *Gesture Wall*'s technological foundation is its use of the four "Fish" sensors, which are described in more detail in Section 4-2. The viewer acts as the transmitter, radiating RF energy through his/her entire body. The four "buds" are the Fish receivers that integrate the transmitted energy from the body in an inverse cubic relationship. Thus the part of the viewer's body that is closest to the receivers, i.e. the hand, will dominate the statistics. A few problems occurred when the participants kept their hands close to their bodies, causing the readings to be biased towards the center of body mass.

Company	Model Number	Equipment
IBM	PC750	(2) Computers, 133MHz Pentium, 128 Mb RAM
---	----	Fish sensor (1 transmitter triangle, 4 transmitter "buds" and calibrator) developed at MIT Media Laboratory
Kurzweil	K2500	Rack mount sampler, 64 Mb RAM, Fujitsu hard drive
Mackie	1202	12 channel audio mixer
Samson	Servo 150	Studio amplifier, 75W stereo
KRK		100W speakers (2)

Table 7-2. *Gesture Wall* equipment list.

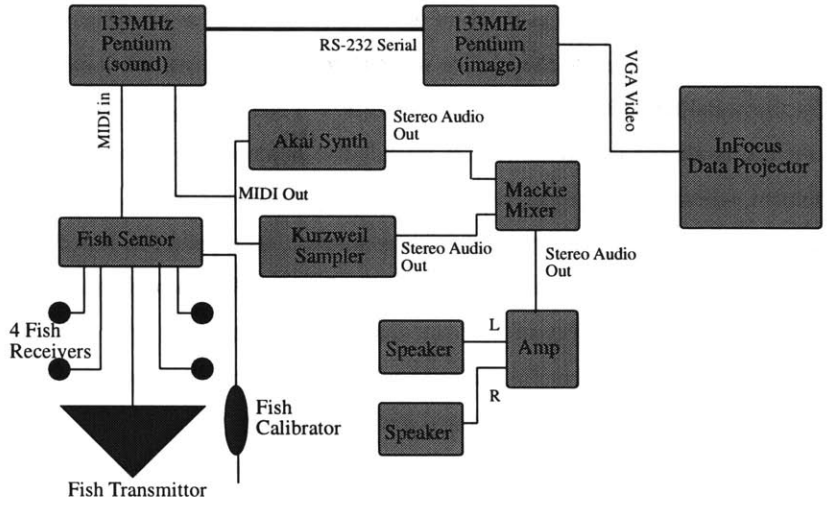

Figure 7-10. *Gesture Wall* system diagram.

Joe Paradiso developed the hardware required for this application and there were a few application challenges to resolve. First, the viewer's transmissive properties were a function of what type of shoe soles were between his or her body and the copper triangle. Wooden-heeled dress shoes would conduct the signal differently than rubber-soled sneakers. Therefore, a calibration unit was developed to allow the participant to register the conductivity of his/her shoes. Before the viewer was allowed to perform on the *Gesture Wall*, he first had to place his hand on the calibrator for a few seconds. Unfortunately, different sized - height and weight - people will produce different results with the fish sensor, and there was no easy way to properly calibrate each viewer for size during the exhibition. It was decided that an averaged sized person would serve as the calibration model for the entire exhibition. This was clearly a compromise that adversely affected the experience for the very short and large viewers, however sufficed for the majority of participants.

The system components of the *Gesture Walls* are outlined in Figure 7-10, consisting, except for the fish sensors, of off-the-self Pentium computers (133MHz), MIDI cards, sound synthesizers, and samplers.

7.6.2 Implementation

The software for the *Gesture Walls* uses a fundamentally different approach to image transformation than the *Melody Easels*. Considering the qualities of interaction with this work - the free-formed motion of arms - suggested a correspondingly different visual metaphor. I decided to consider an image to be comprised of colored "dust" that, when in place, shows the original moving image. This is meant to be a playful re-interpretation of the image as a static data structure, where, normally, the rectangular shape of a video is maintained. Rather than having each row-and-column pixel remain fixed at its assigned spatial coordinate, they are allowed to move about the screen in a well defined behavior that is driven by the viewer's interactions. This, in my opinion, corresponded well with the interface space and created a novel scenario, in which the viewer is brushing away these bits of movie "dust" through the screen space. Again, we used the same notion of multiple video sequences that form layers below one

another, with the viewer's "goal" to reveal as much of the underlying image as possible. Once the top level is mostly cleared away, the bottom level will rise to the surface while another video sequence is loaded from the hard drive.

The interactions and behaviors of the different *Gesture Walls* are slightly different for each unit, however the underlying structure is almost identical. This is intended to demonstrate the utility of a parametrically based interactive environment, where the minute alterations of basically the same systems will produce very distinct cause-and-effect relationships between the viewer and the environment. Furthermore, as noted in the following sections, the behaviors that these autonomous particle agents show are also varied from one system to the other. I nicknamed each *Gesture Wall* according to the quality of behavior that I saw within the system as it played out its set of very localized rules. Each sub-section title contains its corresponding nickname. In my opinion, it would be difficult to achieve this diversity with a finite-state-machine-based interactive work.

For the exhibition, I developed three different interactive visual software behaviors for the five *Gesture Wall* stations. Thus two of the stations doubled the same behavior.

7.6.3 Gesture Wall #1 – Scatter

The visual concept for the first *Gesture Wall* was that the pixel elements that comprised the video image were made of dust that blew about the screen in reaction to the gestures of the viewer. As the participant movs his or her hand, the dust elements which are spatially near the viewer's hand are brushed along in the same direction as the motion. The more the person moves his/her hand, the more pixel dust is floating around the screen, creating an abstracted image of the original. Gradually, the dust settles back down to its original configuration and the original image is once again seen.

However, the challenge was to define a system that produced the flowing characteristic fast enough to be computed in real-time. Furthermore, it was important that the visual environment autonomously re-assembled itself so that all of the dust pixels weren't scattered off the screen. A simple compromise solution was developed that balanced between a particle system and a simple spring system. Each pixel is considered an independent particle, initialized to be at its "proper" place in the spatial coordinates. Let us define two vectors that contain the x and the y spatial coordinates of P image pixel particles:

$$\mathbf{x} = (x_0, x_1, x_2, x_3, ..., x_{P-1}) \tag{7-27}$$

$$\mathbf{y} = (y_0, y_1, y_2, y_3, ..., y_{P-1}) \tag{7-28}$$

furthermore let us define the current velocity of two vectors as well:

$$\Delta\mathbf{x} = (\Delta x_0, \Delta x_1, \Delta x_2, \Delta x_3, ..., \Delta x_{P-1}) \tag{7-29}$$

$$\Delta\mathbf{y} = (\Delta y_0, \Delta y_1, \Delta y_2, \Delta y_3, ..., \Delta y_{P-1}) \tag{7-30}$$

With these velocity vectors, we can update the spatial coordinates of each particle at each time step t:

$$\mathbf{x}[t] = \mathbf{x}[t-1] + \Delta\mathbf{x}[t-1] \tag{7-31}$$

$$\mathbf{y}[t] = \mathbf{y}[t-1] + \Delta\mathbf{y}[t-1] \tag{7-32}$$

which will force the image particles to fly along the direction indicated in the two velocity vectors. In a frictionless environment, the particle would forever travel in this direction. However, in this work, this would mean that the

image pixels would all fly off of the screen. So we need to define a "friction" force that impedes the velocity of a particle. This is approximated by multiplying the velocity vector by a constant, valued between 0.0 and 1.0, at each time step:

$$\Delta\mathbf{x}[t] = \beta\ \mathbf{x}[t-1] \tag{7-33}$$

$$\Delta\mathbf{y}[t] = \beta\ \mathbf{y}[t-1] \tag{7-34}$$

where β is a "friction constant" determining the rate of the slowdown. Therefore, every particle will gradually slow down in the direction of its motion. However we would like that the particle returns to its initial place of origin, therefore we need to define a "spring force" vector that pulls the particle back to its origin:

$$\mathbf{s}_x = \mathbf{x}^0 - \mathbf{x} \tag{7-35}$$

$$\mathbf{s}_y = \mathbf{y}^0 - \mathbf{y} \tag{7-36}$$

where the vectors $\mathbf{x}^0$ and $\mathbf{y}^0$ are the initial spatial coordinates of the particles. So if we combine this with Eqs. 7-33 and 7-34, we get:

$$\Delta\mathbf{x}[t] = \beta\ \mathbf{x}[t-1] + \gamma\ \mathbf{s}_x[t-1] \tag{7-37}$$

$$\Delta\mathbf{y}[t] = \beta\ \mathbf{y}[t-1] + \gamma\ \mathbf{s}_y[t-1] \tag{7-38}$$

where γ is a scalar value between 0.0 and 1.0 and represents the amount of "pull" this virtual spring has. The higher the constant, the "tighter" the spring, keeping the pixel particle close to its origin.

However, so far, this illustrates a closed system as there is no input from the interactive viewer. Keeping the loosely-based physical model in mind, the user is considered as a force matrix that puts these particles into motion according to his/her gestures. From the Fish interface, we get three degrees of spatial freedom, *x*, *y*, and *z* (depth). We will represent the user as a three dimensional vector:

$$\mathbf{u} = (u_x, u_y, u_z) \tag{7-39}$$

From this vector we can determine the instantaneous velocity in time rather straightforwardly:

$$\Delta\mathbf{u}[t] = \mathbf{u}[t] - \mathbf{u}[t-1] \tag{7-40}$$

We need to map this 3-space vector into a two dimensional space that acts as the forces against the image particles. This will provide additional forces against which the particle will react and is notated as:

$$\mathbf{F}_x = \begin{bmatrix} f_{x0,0} & f_{x0,1} & f_{x0,2} & \cdots & f_{x0,M-1} \\ f_{x1,0} & f_{x1,1} & f_{x1,2} & \cdots & f_{x1,M-1} \\ f_{x2,0} & f_{x2,1} & f_{x2,2} & \cdots & f_{x2,M-1} \\ \vdots & \vdots & \vdots & & \vdots \\ f_{xN-1,0} & f_{xN-1,1} & f_{xN-1,2} & \cdots & f_{xN-1,M-1} \end{bmatrix} \tag{7-41}$$

$$\mathbf{F}_y = \begin{bmatrix} f_{y0,0} & f_{y0,1} & f_{y0,2} & \cdots & f_{y0,M-1} \\ f_{y1,0} & f_{y1,1} & f_{y1,2} & \cdots & f_{y1,M-1} \\ f_{y2,0} & f_{y2,1} & f_{y2,2} & \cdots & f_{y2,M-1} \\ \vdots & \vdots & \vdots & & \vdots \\ f_{yN-1,0} & f_{yN-1,1} & f_{yN-1,2} & \cdots & f_{yN-1,M-1} \end{bmatrix} \tag{7-42}$$

These two matrices represent the x and y forces against a particle at a given (j,i) coordinate in the screen space. Since we know the x and y locations of each of the particles, the force applied to each particle p is:

$$f_x[x_p, y_p] = f_{xy_p, x_p} \tag{7-43}$$

$$f_y[x_p, y_p] = f_{y_{y_p}, x_p} \tag{7-44}$$

which may appear to be complicated at first glance, but is easy to implement in software. In plain terms, we need to index the force matrices according to the current x and y location of the image particle. This will yield the force vector is assigned to the area of the screen where the particle is located. In order to create the assignment of these force matrices, we derive motion from the interactive user vector:

$$f_{xi,j}\begin{cases}\dfrac{\lambda u_z(j - u_x)}{(j - u_x)^2 + (i - u_y)^2} & \sqrt{(j-u_x)^2 + (i-u_y)^2} < D_{\max} \\ 0.0 & \text{otherwise}\end{cases} \quad i = 0...N-1,\ j = 0...M-1 \tag{7-45}$$

$$f_{y_{i,j}}\begin{cases}\dfrac{\lambda u_z(i - u_y)}{(j - u_x)^2 + (i - u_y)^2} & \sqrt{(j-u_x)^2 + (i-u_y)^2} < D_{\max} \\ 0.0 & \text{otherwise}\end{cases} \quad i = 0...N-1,\ j = 0...M-1 \tag{7-46}$$

Here we use the z value of the user's hand to increase the amplitude of the force. Furthermore, the λ is used to provide a constant scaling. With these force assignments, pixels are "pushed" outwards from the place of the user's hand, inversely proportional to the distance of the point to the user's hand. D_{max} is the maximum distance over which this force assignment operates, in order to limit the number of computations required. Sample force vectors are shown in Figure 7-11.

Now that we have the force matrix assignment completed, we can implement these forces into our particle velocity updates. But first let us define a force vector whose elements are the forces for a given particle at its current location, continuing the notation from Eqs. 7-43 and 7-44:

$$\mathbf{f}_x[t] = (f_x[x_0, y_0, t], f_x[x_1, y_1, t], f_x[x_2, y_2, t], ..., f_x[x_{P-1}, y_{P-1}, t]) \tag{7-47}$$

$$\mathbf{f}_y[t] = (f_y[x_0, y_0, t], f_y[x_1, y_1, t], f_y[x_2, y_2, t], ..., f_y[x_{P-1}, y_{P-1}, t]) \tag{7-48}$$

These two vectors, although with a tricky notation, merely index the two force matrices on a particle-by-particle basis according to where the particle is at any point in time. This makes the next step easy, where we add this force to the velocity update step:

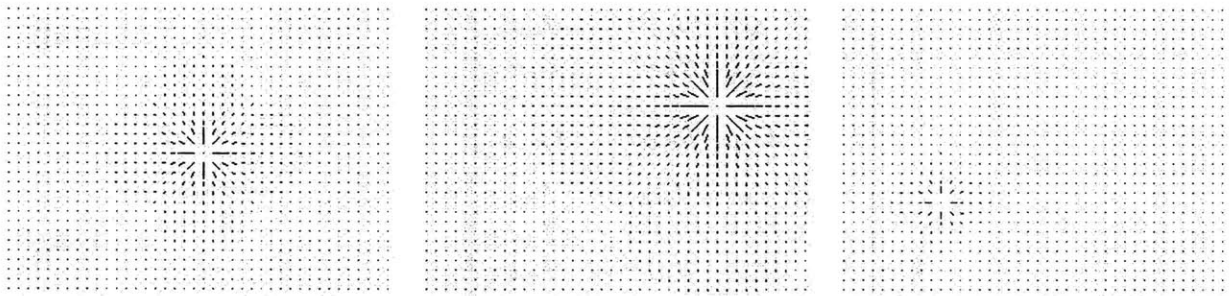

Figure 7-11. Three force matrices used by the particle system. The magnitude of the force is inversely proportional by the distance from the stimulus.

$$\Delta \mathbf{x}[t] = \beta\, \mathbf{x}[t-1] + \gamma\, \mathbf{s}_x[t-1] + \alpha\, \mathbf{f}_x[t-1] \tag{7-49}$$

$$\Delta \mathbf{y}[t] = \beta\, \mathbf{y}[t-1] + \gamma\, \mathbf{s}_y[t-1] + \alpha\, \mathbf{f}_y[t-1] \tag{7-50}$$

where α is used as a scalar multiplier to weight the force input. Therefore this particle system has four conceptual components: momentum, friction, spring force, and applied forces. This set of system components provide for a very rich set of behaviors in the graphical environment. Note that the presence and actions of the viewer are indirectly encoded in the $\mathbf{f}_x$ and $\mathbf{f}_y$ vectors.

The name "scatter" was chosen as the behavior that this interactive environment exhibits reminded me of a group of gnats flying almost suspended in the air. When one swats at the bugs, they scatter according to the direction and strength of the arm gesture, due to air turbulence. However, if one does not continually flair his or her arm, the gnats will regroup themselves, seemingly in the same place as they were originally at.

This entire system is updated for every particle in the system and rendered to the output video buffer, placing the video image pixel at the particle's current location on the screen. Unfortunately, no spatial "averaging" was done where, if more than one particle is at the same location, multiple pixel values are mixed with an even weight. This would have required more computational power than what was available at the time. Therefore, whichever pixel is rendered last, covers up the other pixels at that location. Since there are two image planes, the top and lower level videos, the lower image is spatially fixed – it is a normal video sequence - and rendered first into the video output frame buffer. Then the particle system is computed, updated, and rendered to the frame buffer. If a particle does not cover up a particular area in the screen, it is possible for the user to see the lower level. However, if all the particles are at rest at their origins, then the viewer sees only the top level video.

Figure 7-12 shows some screen captures of this *Gesture Wall*. In these images, taken directly from the computer's frame buffer, the particle system begins in a scattered state and re-assembles itself due to the spring forces that draw the particles back to its original starting point. In the first image, you can see the second video clip "peering" through the gaps on the top video surface. Unfortunately, as this output is highly animated, still frames do not quite portray a good impression of this effect.

7.6.4 Gesture Wall #2 – Sweep

The second *Gesture Wall* type is mathematically very similar to the "Scatter" *Gesture Wall*. In fact, it is identical except that one component of the behavioral system of equations is removed, which is the spring force that brings the picture particles back to their origin. So the interactive system is slightly simpler and if we revisit Eqs. 7-49 and 7-50 as defined above, we get:

$$\Delta \mathbf{x}[t] = \beta * \mathbf{x}[t-1] + \alpha * \mathbf{f}_x[t-1] \tag{7-51}$$

$$\Delta \mathbf{y}[t] = \beta * \mathbf{y}[t-1] + \alpha * \mathbf{f}_y[t-1] \tag{7-52}$$

Here only the momentum, friction, and applied forces are kept in the system. The rest of the mathematics is identical to *Gesture Wall* #1 and therefore will not be repeated here.

What is interesting about this small change of equations is the profoundly different qualitative behavior the system exhibits. Rather than the screen ending into a never ending flurry of image pixels that race around the screen,

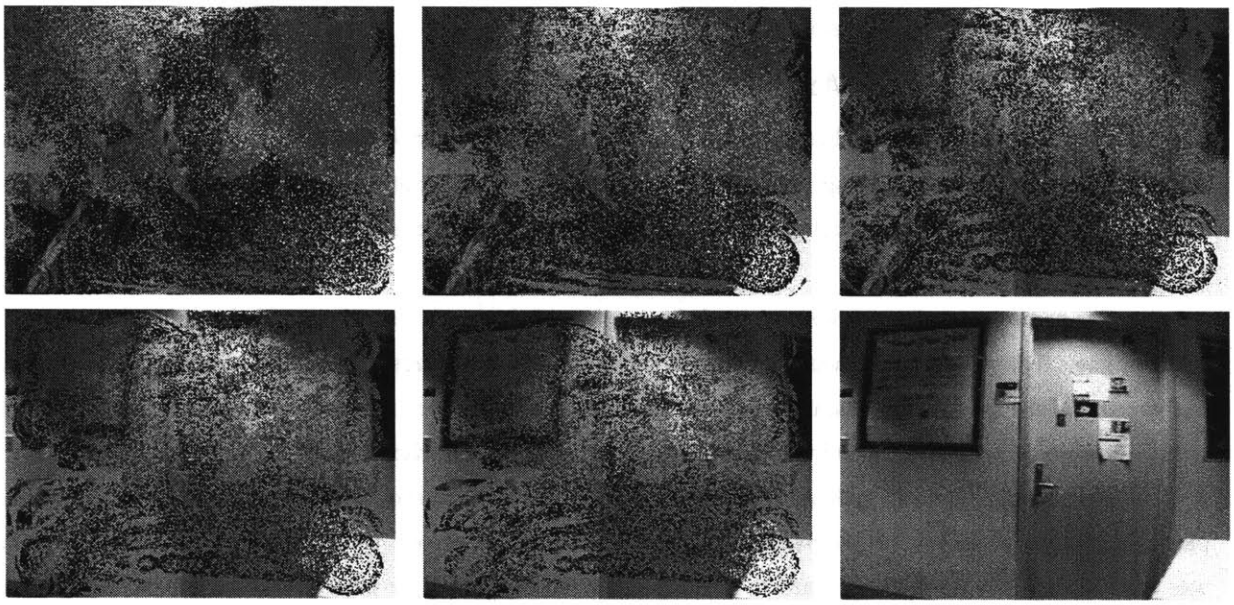

Figure 7-12. Sample output from the "Scatter" *Gesture Wall* as the particle system "re-assembles" itself

this *Gesture Wall* is much more calm. The name "sweep" was chosen as it was fitting to the particle's behavior - the user brushes aside the image particles, as if they were dust on the floor being pushed around by a broom.

Since the only primary forces are stemming from the user him/herself, the cause-and-effect relationship between the participant and the system is more simple than in the *Gesture Wall* #1. In the "scatter" system, the spring force that caused the particles to return home were "invisible" to the viewer and, perhaps, not intuitive as the particles exhibited autonomous behavior which was not related to what the viewer was currently doing. While this led to a more complex and dynamic environment, it was a slightly more abstract form of stimulus-response.

In order to transition from one scene to the other, i.e. raising the lower video sequence to the front, a simple function was used:

$$d[t] = \sum_{p=0}^{P-1} \sqrt{(x_p[t] - x^0{}_p)^2 + (y_p[t] - y^0{}_p)^2} \qquad (7\text{-}53)$$

which merely sums up the distances of the particles from their original initial places. When this distance metric is greater than some threshold, the particles of the top layer are gradually moved off screen, leaving the back video sequence in full view. The particles are reset and another background layer is loaded in from the hard drive.

Figure 7-13 shows some examples of the *Gesture Wall* #2 software. Note that the top layer image is solid initially. As the viewer, with his/her hand, "brushes" away the top layer particles, the bottom image becomes ever more visible. When the bottom image is completely uncovered, it is made to be the top layer, allowing for the viewer to "sweep" away this image to discover yet another video sequence, and so on.

Figure 7-13. Six images from the "Sweep" *Gesture Wall* output

7.6.5 Gesture Wall #3 – Swarm

The "Swarm" *Gesture Wall*, takes its name from the behavior it produces. Here the pixels, rather than being scattered or brushed away (repulsive behavior), are attracted to the users motion. As the viewer moves his or her hands, the picture elements that are close to the place of motion fly towards the user's gestures. When the motions cease, the image particles gradually fly back to the place of their origins.

The mathematics of this behavior is also very similar to *Gesture Wall* #1, but there are some notable differences. The complete interactive system is still defined by these iterative equations:

$$\mathbf{x}[t] = \mathbf{x}[t-1] + \Delta\mathbf{x}[t-1] \tag{7-54}$$

$$\mathbf{y}[t] = \mathbf{y}[t-1] + \Delta\mathbf{y}[t-1] \tag{7-55}$$

$$\Delta\mathbf{x}[t] = \beta * \mathbf{x}[t-1] + \gamma * \mathbf{s}_x[t-1] + \alpha * \mathbf{f}_x[t-1] \tag{7-56}$$

$$\Delta\mathbf{y}[t] = \beta * \mathbf{y}[t-1] + \gamma * \mathbf{s}_y[t-1] + \alpha * \mathbf{f}_y[t-1] \tag{7-57}$$

The momentum, friction, and spring forces are still the same as in *Gesture Wall* #1. The only difference is in the assignment of the force matrices, $\mathbf{F}_x$ and $\mathbf{F}_y$, and the relationship between the user interface and the cause-effect mappings.

Let us return to the force matrix assignment and redefine the relationship between it and the user salient vector. In this system we need to bring the particles *inwards* towards the hand rather than outwards, as was done previously. This can be accomplished by:

$$f_{xi,j}\begin{cases}\dfrac{\lambda u_z(u_x - j)}{(j - u_x)^2 + (i - u_y)^2} & \sqrt{(j-u_x)^2 + (i-u_y)^2} < D_{\max} \\ 0.0 & \text{otherwise}\end{cases} \qquad i = 0...N-1,\ j = 0...M-1 \tag{7-58}$$

$$f_{yi,j}\begin{cases}\dfrac{\lambda u_z(u_y - i)}{(j - u_x)^2 + (i - u_y)^2} & \sqrt{(j-u_x)^2 + (i-u_y)^2} < D_{\max} \\ 0.0 & \text{otherwise}\end{cases} \qquad i = 0...N-1,\ j = 0...M-1 \tag{7-59}$$

which simply changes the sign of the force vector to point inwards towards the user. Particles that are further away from the hand are drawn in more slowly than pixels closer to the user. The constants λ and D_{max} serve the same function as defined in Eqs. 7-45 and 7-46 however need not be set to the same values. Likewise the transition criteria is merely inverted: when the sum of the distance of the particles relative to the user's hand falls below a threshold, the top video sequences complete collapses and the back video sequence becomes the top one.

Once again, this demonstrates the vast utility of using parametric systems rather than state spaces for interactive environments. Here we only need to change the sign of one of the continuous parameters of the system in order to get a radically different qualitative behavior of the entire environment. From the same equations that gave an impression of a group of gnats we now get a impression of a swarm of predators that descend upon the user's hands, all with a simple sign change!

Figure 7-14 shows six images from the Gesture Wall #3 output. When the viewer makes hand motions, all of the particles "swarm" to the location where his/her hand is. The particles will continue to follow any hand gestures that the participant may make, streaking across the screen to catch up with the viewer. When the participant stops making motions, the particles are gradually "pulled" back to their starting place to wait for the next motion that is sensed, causing the particles to "swarm" once again towards the viewer's hand.

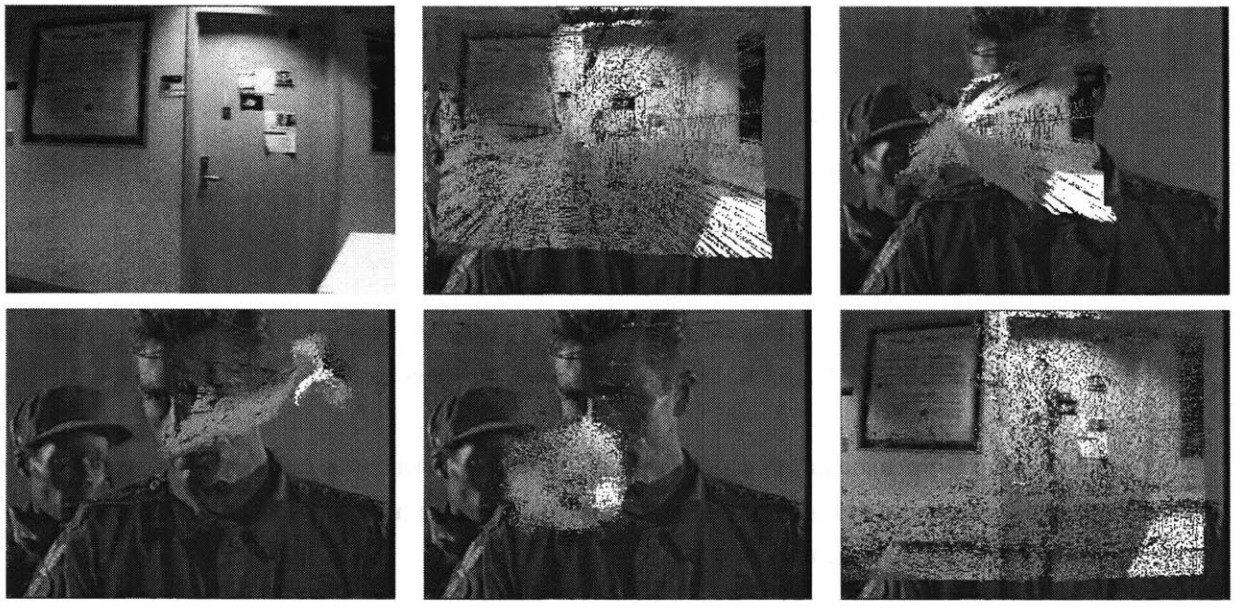

Figure 7-14. Six frames from the "Swarm" *Gesture Wall* output..

Chapter

8 Improvements to the Gesture Walls

This chapter reviews the design and implementation of the *Gesture Wall* environment as it was developed for the Lincoln Center premiere in July 1996. Since the opening of *The Brain Opera*, several fundamental modifications have been made to this interactive environment in order to address a few issues that arose while both observing the audience's participation and receiving feedback from colleagues.

This iterative design approach is typical for computer based interactive art. Often one premieres a work, receives feedback and suggestions, updates the design and implementation, demonstrates the new work, receives further feedback, etc. As this field is relatively new, it is difficult to get it "right" the first time. With each design iteration, it is possible to correct weak areas of the experience based on the reactions of the audience. Since the development cycle for interactive environments tends to be long, the artist/engineer often looses objectivity in regard to the experience. By revisiting older designs after some separation from the development cycle, the artist can approach the work again with freshness and new vigor. Furthermore, as the technology improves and yields higher performance systems, the artist/engineer revisits older work in order to take advantage of the new possibilities.

8.1 Problem Identification

After talking with both *Brain Opera* audience members and fellow interactive system programmers, several key criticisms of the *Gesture Wall* repetitively arose. While, on the whole, everyone enjoyed the system in the Lincoln Center version of the environment, many believed that additional improvements could be developed and tested. In general, the *Gesture Wall* presented the familiar problem of how to instruct participants on how to use the system.

To me, the largest problem with the *Gesture Wall* stems from the literal spatial mapping of the participant's hand to the visual processing "effect." Normally, in face of such a general and wide public, direct cause-effect mappings are appropriate in order to make the learning process accessible to those not familiar with interactive art. However, although the mathematical systems performed this one-to-one relationship, there were sensing "errors" with the Fish input. Due to the compromise of not performing a "full" calibration of the participant (i.e. we only accounted to variances in shoe conductivity and not variations of body size and height), the data being generated at the input section was not a reliable description of the hand of the user. Furthermore, as the Fish sensor measures merely the center of mass of the body as an inverse function of distance, when the participant did not reach forward with his/her hands, the measurements were biased strongly towards the center of his/her torso. It is important to realize that these errors were not a result of "noise" and thus could not be simply filtered away. These hand position estimates were the result of a clean signal that was correlated to several unknown variables. These unknown factors were due to a lack of sufficient knowledge concerning the viewer, i.e. is the person reaching properly out, how tall is he/she, how heavy, etc.? In order to derive accurate sensing of the viewers, we would have needed additional sensing information that was not available.

The net effect of these sensing issues was unfortunately perpetuated through the entire interactive system. Thus this forms a useful example for the dependency of each sub-system on the previous section within this model of interactivity. Although all of the hardware development, input software, mapping functions, and output reconstruction were flawless, the system did not give the appropriate correspondence between the viewer's real hand and the environments "virtual hand." When standing in front of a large video projection which was, supposedly, under the control of one's hand, it became immediately apparent that the real and the virtual did not match up. The design choice in this case, on my part, to literally map spatial positions from the physical world into the virtual world was a poor decision. The reason why this is not an example of a rich and varied experience is because the tension between both extremes, a literal mapping versus a more abstract one, were *not* under the control of the system. The almost constant denial of a harmonious mapping was due to insufficient sensing data; therefore it was wrong to allude to a one-to-one mapping when it was, for the most part, impossible to achieve. The solution, which will be introduced in Section 8-3, was to not allude to such a precision of control and to further abstract the representation of the user from a literal into a more hazy format.

Another problem stemmed from the reconstruction design choice, when, as described in Chapter 7, we opted for a modified particle system with a number of virtual forces some of which were derived from the viewer's motion while others were internal to the physical-based model. The desired effects of a swarming and floating set of pixel particles that flew apart and recombined were not achieved due to the complexity of the visual output. While the visual effect looked impressive on the development computers' monitors, the LCD projection screens that were used in the exhibitions were not capable of giving a bright image with enough contrast, due to ambient light in the *Mind Forest*. Thus the video effect just looked very fuzzy and not at all clear to the participant. Considering the complexity of the abstract form of interaction – from hand motions to virtual particle forces – a very clear and crisp visual output would have been required. As the equipment for the *Brain Opera* had already been fixed, we could not simply buy brighter and higher-quality LCD video projectors, so I decided to fix the interactive visual environment instead.

Another subjective criticism that I found was that the visual environment did not "fit" the interactive music environment for the same *Gesture Walls*. The Tod Machover composition encouraged, in my opinion, the viewers to make bold, strong motions with their hands when first encountering the experience. Gradually, after this initial visceral exploration, the viewers would slow down their motions to investigate the more subtle nuances of the interactive music. However, with the current software for the visual response, such initially strong motions seemed to confuse people as the video projection would be transformed into a crazy flying set of image pixels. While observing several unbiased audience members try the *Gesture Wall*, it was my desire to take a new approach that had the same rewards as the music section.

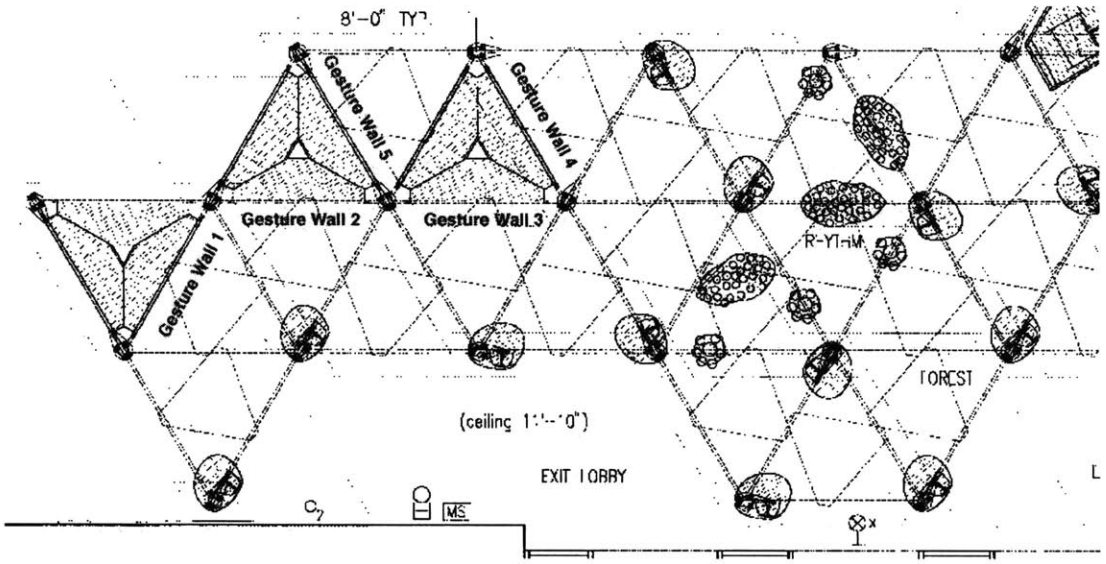

Figure 8-1. Overhead view of a portion the *Interactive Mind Forest* layout. The five *Gesture Wall* stations are marked in the upper left corner.

One additional goal was to make the responsiveness of the *Gesture Walls* a bit quicker. The original design produced images at about 7 FPS which is a little on the slow side. I had hoped to yield frame rates in at least the double digits, in order to keep the visual experience as smooth as possible.

8.2 Opportunities for Community

These three main points of contention with the *Gesture Wall* design and implements all apply to a single user environment. However, the *Gesture Walls* have a wonderful physical structure devoted to this experience with the projection screens, each for one *Gesture Wall* system, forming a nice, nearly smooth surface. Furthermore, the five stations form an almost continuous circle that wraps all the way along. Figure 8-1 shows an structural overview of the layout of the *Brain Opera*, with the five *Gesture Wall* stations to the left.

Such a nice structural layout appears to be a ripe area for experimentation of shared, multi-user environments. As described in Chapter 5, these models of interactivity can easily be geared towards collaborative environments. Here everyone would receive a common experience that is not only based on the computational cause-effect mappings between the single user and the system, but rather is a function of how everyone is interacting as a community. Since the projection screens are aligned next to one another, it also presents the opportunity to create a larger virtual "screen" that bridges the entire section. In this way, one viewer can see what his/her neighbor is doing, on the periphery, while continuing with his/her local interactions.

Therefore, this chapter will introduce a new approach to the *Gesture Walls* in which the environment is a collaborative and distributive interactive installation that uses the entire set of five interactive stations in order to visualize the collective body of people. In addition, I wished the group interactions to be additive in nature, meaning that whatever contribution one participant would make could be reinforced or destroyed by others.

8.3 New Gesture Wall Design

A new design has been implemented as part of this thesis research which attempts to address the above concerns using the general framework that has been outlined in this document. In general, the new design goals are to provide for a multi-user distributed environment that uses one overarching system for the control and coordination of all individual stations. This is to say that each of the five *Gesture Wall* stations runs the exact same program with the same interactive mappings. However, they all share one data set and are part of one thematic idea that is visualized through their interactions.

The visual idea is to link the continuous quality of a particle flow with the *Gesture Wall*'s even structural design, creating a single unifying visual output that bridges each of the individual stations. This particle system is different than that which is described in Chapter 7 in two ways. First, there was no attempt at giving a particle an "initial" or "home" spatial location on the screen. Second, there is no video sequence running but rather the particles are given a color assignment that is detailed below.

Only the interactive graphics system has been modified in this revision. There were no changes to the interactive musical system.

8.3.1 Use of Fish sensors

The fundamental problem, in my opinion, with the Lincoln Center *Gesture Wall* implementation was the information that was obtained from the Fish input sensors. For several reasons, it was not possible to take the data at face value, as there were many additional user variables, e.g. viewer's body size and height, that caused unwanted distortions of the data (as discussed in Section 8-1). If two viewers, A and B, pointed to the exact same region on the screen, the input data from A and B could be extremely different. It was not a valid assumption that we could take absolute (x,y,z) information from the Fish sensor.

I decided that since absolute information may be prone to error, it might be wiser to use differential information. A person moving his/her hand to the left would still always cause the corresponding x value from the Fish sensor to decrease, so the sign of the difference is correct regardless of whether the amplitude information is true. However, it was still difficult to work in a pure relative coordinate system, as the visual effect ultimately had to be rendered in some absolute screen coordinate system. Again, the same problem of *where* to apply the visual output was considered. As described in the following sub-section, it was necessary to continue to use some absolute measurement of the user's hand within the Fish interface.

8.3.2 User Representations

In this updated version of the installation, we wish to make the representation of the interactive participant more robust. I have decided upon two major forms of user representation: a slowly adapting region vector and a local

temporal differential vector. Let us consider the 3-tuple vector that is arriving from the Fish sensor as a discretely sampled signal:

$$\mathbf{u}[t] = (x[t], y[t], z[t]) \tag{8-1}$$

If we define another vector to represent position of the user's hand in an iterative manner:

$$\mathbf{l}[t] = (1.0 - \beta)\mathbf{l}[t-1] + \beta\,\mathbf{u}[t] \tag{8-2}$$

where β is an integration constant between 0.0 and 1.0. If β is set to a value close to 1.0, then vector **l** will be quick to react to changes at the Fish input. However, if β is set to a low value closer to 0.0, this vector will move very slowly, making it very robust against both Fish noise and quick jittery hand motion.

Second, we would like the interactive control of the environment to be based on differential calculations of the Fish sensor. Therefore at each time t we calculate:

$$\dot{\mathbf{u}}[t] = \mathbf{u}[t] - \mathbf{u}[t-1] \tag{8-3}$$

which is simply the subtraction of each Fish location estimates. This will yield a relative motion vector that has both direction and amplitude.

As we would like to correlate the motions of the particles to the motions of the hand, we need to define two force matrices just as in Chapter 7, describing the forces projected onto the x and y axis. These matrices represent virtual forces that are applied to the free floating particles at every time step. In order to make the force matrix assignment, we take a slightly different approach here then previously done:

$$f_{x_{i,j}} = \begin{cases} \dfrac{\lambda \mathrm{u}_z \dot{u}_x}{\sqrt{(j-l_x)^2 + (i-l_y)^2}} & \sqrt{(j-l_x)^2 + (i-l_y)^2} < D_{\max} \\ 0.0 & \text{otherwise} \end{cases} \quad i = 0...N-1,\ j = 0...M-1 \tag{8-4}$$

$$f_{y_{i,j}} = \begin{cases} \dfrac{\lambda \mathrm{u}_z \dot{\mathrm{u}}_y}{\sqrt{(j-l_x)^2 + (i-l_y)^2}} & \sqrt{(j-l_x)^2 + (i-l_y)^2} < D_{\max} \\ 0.0 & \text{otherwise} \end{cases} \quad i = 0...N-1,\ j = 0...M-1 \tag{8-5}$$

These equations are used to define a force matrix $\mathbf{F}_x[t]$ and $\mathbf{F}_y[t]$ for every time step. However, in order to smooth out the interactive graphical environment, this force matrix is averaged in with it's previous value:

$$\mathbf{\Gamma}_x[t] = \gamma(\mathbf{\Gamma}_x[t-1], \mathbf{F}_x) \tag{8-6}$$

$$\mathbf{\Gamma}_y[t] = \gamma(\mathbf{\Gamma}_y[t-1], \mathbf{F}_y) \tag{8-7}$$

The non-linear function named $\gamma(\ )$ is to make sure that there's at least a minimum absolute value for each force value, otherwise there is no change:

$$\gamma(\varphi, f\ ) = \begin{cases} (1.0-\alpha)\varphi + \alpha f & |f| \geq \varepsilon \\ \varphi & \text{otherwise} \end{cases} \tag{8-8}$$

where α is an integration constant that regulates how quickly the force will adapt to the new value. ε is a constant which is the minimum value the new force must be in order to be averaged in. This is to prevent the force matrix from going to zero when no interactive forces are being applied by the participant.

Taken together these equations create the force matrices that are applied to the participle system, setting them into motion. If a viewer placed his hand at a particular point in the (x, y) image plane and made some hand

motions, the "trace" of this gesture would be captured into the force matrices, averaged in with all of the other motions that were previously performed in this area of space. All forces are retained in the system until other motions are made, due to the above non-linear minimum threshold function. This had the corresponding effect of leaving a "ghost" of the gesture in the system until contrary motions were made. Should the participant disengage from the environment, the system would still nevertheless continue to present this gestural trace forever.

8.3.3 Reconstruction

The challenge of the updated *Gesture Wall* was in the creation of a continuous visual output, where the visual reconstruction of the interactive environment is distributed over the entire set of five stations. While the computation was performed on each of the machines independently, it must appear to the viewer that the entire *Gesture Wall* environment is smooth and continuous. The behaviors of the graphical particles must scale well over all of the stations, allowing the audience to stand back from the environment and take in the dense set of interactive traces that were developing. Furthermore, the qualities of the reconstruction should be in correspondence with the thematic style of the new work.

I decided that the particle system should "bridge" each of the individual systems, in order to make smooth bands of colorful streams. When the image material floated "off screen" on one station, it would appear at the other side of its neighbor's screen to continue on its floating journey. The same visual material would be shared by everyone, with each individual providing the virtual forces that push the particles along. If one participant makes left to right motions, all of the image particles will be, eventually, pushed off screen and into his/her right-hand neighbor's environment. This neighbor can "command" these particles as he/she wishes, either reinforcing the left to right floating with similar hand motions or even contradicting the other person's efforts by making contrary motions.

The meta-behavior, i.e. "cooperative" or "destructive", of the interactive community would reveal itself through the resulting interplay between each of the individuals at the stations. For example, if one participant was making concerted hand motions from left-to-right, pushing the image particles off to his/her right hand neighbor who was making contrary right-to-left motions, the resulting particle flow behavior would bounce back and forth between the two stations. This discord of interactive gestures is made saliently apparent to the audience members who were standing back a little, observing the actions of the two participants.

In order to accomplish these goals, the local area network was used to connect neighboring *Gesture Wall* stations, using TCP/IP software sockets in order to create communication links. Figure 8-2 shows the network topology of the *Gesture Wall* network. When the software starts, it tries to connect with the IP address of its right-hand neighbor and waits for a connection from the left-hand side. The right and left communication channels operate over different socket ports, making the software protocol very simple. All that is necessary is to read and write the internal data structures to the socket port, as no other communication protocol is needed. Furthermore, as the particle data structure is relatively small, communication latency and bandwidth considerations were not a problem.

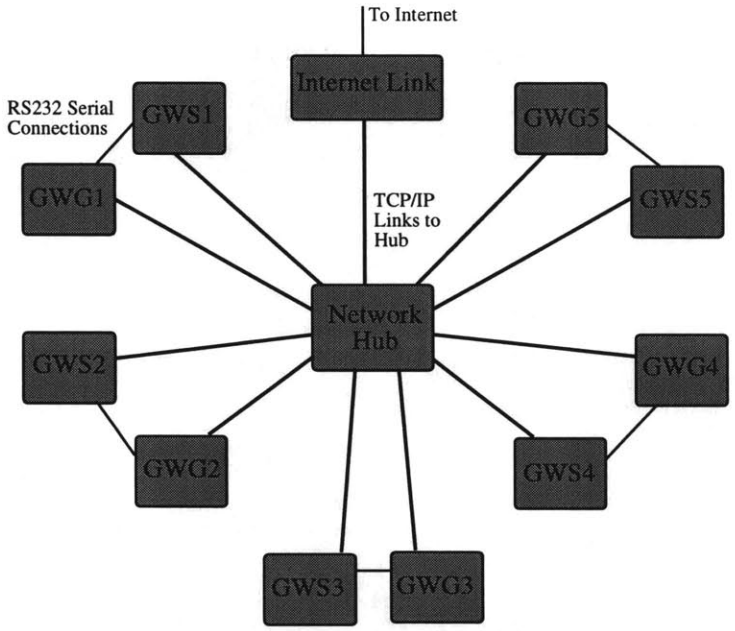

Figure 8-2. Network topology of the *Gesture Walls*. Each computer is connected to a centralized hub, linking each station to one another. A RS-232 serial connection is used for communication between the sound and graphics systems.

When a particle floats off the screen, a four-dimensional particle description vector is transmitted to the neighbor. This vector consists of the particle's position and velocity at the point when it left the screen. The neighboring station, upon receiving the particle, adds this new particle to its internal list of particles that it manages and visualizes. Since the position and velocity information is transmitted between neighboring machines, the particle's apparent motion is smooth and continuous. The newly transferred particle is now subject to the interactive forces that are made by the neighboring participant, either to be reinforced along its current path or diverted in another direction.

The particle system's update, render, and transfer operations are performed at every frame. The mathematical implementation of this new updated *Gesture Wall* is the same as that which is discussed in Chapter 7. Again, through relatively small alterations in the interactive system, large behavioral changes can be observed. In the modified *Gesture Walls*, all that was altered was the removal of the spring forces in the reconstruction model and the addition of a transmission operation. This is meant to be a strong argument for this thesis' proposed model of interaction, whereby a significantly different end-user experience can be formulated through subtle manipulations of the systems.

Six output images of the modified system are shown in Figure 8-3. As the implementation was performed outside of *The Brain Opera* as a separate thesis project, the software was run within a different environment. The images illustrate three main components of the system. The first column of images shows the particle flow over a set of three computer monitors. As made tangible through the photos, the transmission of image particles across the data network between machines allows for a continuous "macro-texture band" of image material that spreads over multiple computers. The photos in the second column are medium close-ups of adjacent monitors, showing the

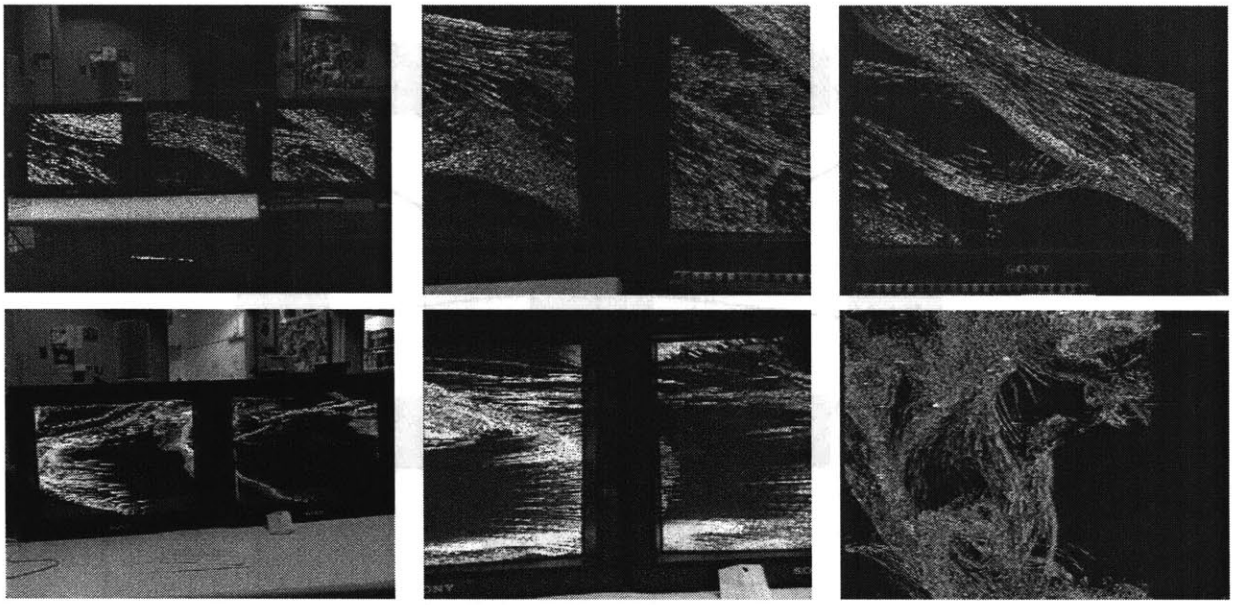

Figure 8-3. Six images from the particle system output.

continuity of image material over the computers. The last column is a set of close-ups that represents the fine "micro-textures" that the system is capable of producing.

As clearly demonstrated in this figure, the modified Gesture Wall system invites the viewer to approach the work at a number of scales - from a macroscopic overview of the overwhelming particle "formations" to the small details of floating colorful material. This is intended to elicit a series of subsequent inspections on the part of the audience, as first they would receive a macroscopic view of the environment, be entranced in by the smooth image flow. After watching passive the environment, each individual would be drawn towards trying out the interactive stations him/herself in order to concentrate on the microscopic details.

8.3.4 New Color Mappings

As one of the problems with the initial design of the *Gesture Wall* was with the poor image contrast by the LCD projector onto the translucent material, stronger primary colors was used in the modification. As the particles were rendered onto a black background, the foreground colors were much easier to distinguish. Second, as the environment no longer used video sequences as it's imagery, there was no need on the part of the viewer to have an concrete cinematic experience, i.e. there are no video sequences playing on the screen. This allowed me to use a more abstract color representation.

The color "temperature" of a particle was directly proportional to its velocity. This borrowed its color metaphor from the notion of a moving body heating up according to its speed. Slow particles used deep blue colors while faster moving particles tended to red colors. The fastest particles were given white-ish coloring. This simple mapping of velocity to color is relatively consistent with people's expectations and is hoped to be intuitive.

8.4 Inter-user interactions

Although the individual participant has direct control over his/her immediate environment in the Gesture Wall station, they are contributing to an overall particle stream that bridges all of the participants together. With the simple visual concept of a particle system, the notion of a stream that flows, either lazily or rapidly, through the entire group makes for a sense of community.

How that community operates together as a whole is made tangible when an audience member steps back and observes the entire *Gesture Wall* environment. The presence of the interactive participants is revealed through the behavior of this particle stream, while the quality of the flow indicates how the viewer interacted with the system. If the individual participants were "cooperative", in the sense that they all made similar motion patterns, the particle system will gradually evolve into a very harmonious stream that circulates eternally throughout the set of Gesture Walls. Should some participants be "uncooperative", making motions that are contrary to the collective flow of particles, the outside observer will see a steady stream of particles that are destroyed when reaching the disruptive force matrix.

Furthermore, as the system "retains" the force matrix, until the next participant uses the station, the shadows of each of the participating viewers remain indefinitely. This contrasts with many other interactive works in which the system is either static when no participant is present or resets itself when the next viewer engages the work. As the force matrix, in its mathematical definition, continually averages in new gesture information with old data, the flow of particles does not merely represent the interactions of a single participant, but is an evolving layering of gestural contributions. Such an averaging mathematical system also contrasts with other forms of contributive interactive works in that there is no clear remnant of the original, only continual transformation.

Lastly the inter-person interactions are significant at a number of scales. When an individual is using the interactive environment, he is receiving a subjective view into a global state. This is to say that the individual, when engaging the system, looses all sense of objectivity because she is operating at a much more local scale. Although the Gesture Wall environment is a continuous flowing surface, one can either be a subjective contributor or an objective observer, but not both at the same time. It is a little like viewing large format traditional artworks, where either one can look close at the details, but not be able to view the entire work, or stand far aback to receive an overview of the work, losing perception of small details. This duality of engagement – an active engagement in the interactivity or the passive engagement of observation – brings the viewer into one or the other role. This is pleasing to me as, according to some of the theoretical discussions in Chapter 1, the interactive participant should not be given complete control over the environment.

Chapter

9 Evaluation, Future Directions, and Conclusion

This chapter serves as a brief review of the theory, implementation, and test cases of this proposed model of interactive art. To reiterate the obvious, it should be said that the model detailed in this thesis is only one possible approach to creating computer based interactive artworks. There has been a wide assortment of successful interactive works over the past decade - many of which are described in Chapter 2 - that do not use this model at all. Furthermore, success or failure in this field is difficult to ascertain, due to the highly subjective nature of art. However, I wish to take this opportunity to describe participants' reactions to the test cases.

To review the goal of the thesis, I have attempted to create a parameter-based model of interactive environments. These environments are most appropriate for the communication of abstract, non-narrative thematic content related to an exposition of process. An exposition of process is defined by the qualities of a computational system that presents a continual unfolding of an aesthetical experience to the viewer. Overall, the system is more concerned with a communication of qualities of being, rather than a literal portrayal of thematic content. This is to allow the viewer to subjectively experience the environment, rather than keeping him/her at an objective distance.

The system model is broken into three major components, each with a formal set of mathematical representations and operations. The first section abstracts the presence of the viewer into a user representation that is called the salient vector. The salient vector is a multi-dimensional, continuous parameter vector, whose exact semantic definition is the task of the artist/engineer. The salient vector attempts to describe the most significant components of the physical world in which the viewer exists. Possible salient vector parameters, as suggested in Chapter 4, include optical flow, visual color and texture, and non-intrusive sensing devices such as the Fish sensor. This salient vector is all that the interactive environment knows about the viewer and, therefore, it is critical that the *correct* qualities are derived. Of course, "correct" is defined by the particular piece that is to be developed and must be decided on at the beginning of the project development.

The next system component is the transmission of this salient representation of the local interactive environment to one or more remotely located environments. Furthermore, salient vectors are received from these remote stations and mathematically included with the local user representation. Depending on the number of connected users, an interactive community can develop spontaneously where each individual is not only interacting with the local content but with the shadows of the others.

Finally, the salient vector, now including any remotely connected viewers, is fed into a reconstruction model that presents an "output" to the local viewer. This creates a feedback to the actions of the collective group of participants, illustrating the cause-effect relationships between the local viewer, the remotely connected participants, the input-to-output parameter mappings, and the final recontextualization of the digital system into the real-world. Several examples of reconstruction styles were presented in Chapter 6, including both synthetic computer graphics, image processing, and tangible/ambient output.

9.1 Forum for Evaluation of Test Cases

This model was applied to several test cases in this thesis project in order to evaluate the validity of the proposal. These works were made part of Tod Machover's *The Brain Opera* project as interactive stations within the *Mind Forest*. A total of eight systems were developed for the Melody Easels and Gesture Walls, undergoing several revisions throughout the development and exhibition period. I developed the interactive graphical systems under the guidance of Sharon Daniel, visual direct of the entire *Brain Opera*.

These interactive environments have been presented at five different venues to date as of August, 1997: Lincoln Center (NYC), Ars Electronica (Linz, Austria), The Electronic Cafe International (Copenhagen, Denmark), the Yebisu Garden Center (Tokyo, Japan), and the Kravit Center (West Palm Beach). It is most likely that *The Brain Opera* will find a permanent home in a museum within the next few years so that audiences will be able to continue to enjoy the unique experience that is offered. Thus, these test cases have reached and will continue to reach a large public, providing a good benchmark for evaluating the success or failures of my work.

Overall, the public and critical response to *The Brain Opera* has been very positive. It is the largest interactive work to date with over 50 interactive stations, offering many diverse types of experiences. It has proven to be reasonably robust enough to take on such a world tour without any major mechanical or technical problems. This is a credit that everyone on the *Brain Opera* team deserves. Without a doubt, this work will stand out in history as one of the pioneering examples of large-scale interactive art and entertainment. I hope that other such efforts in large-scale exhibits will continue in the future by my colleagues at the MIT Media Lab and elsewhere.

9.2 Evaluation of the Melody Easels

The *Melody Easels*, in my opinion, were the most successful of the test cases for many important reasons. Overall there was a nice consistency between all of the major components of the installation, creating a nice thematic arch that covered the entire experience. First, the structural physical design of the Easels was both inviting and charming, looking like three suspended cauldrons that could have been from the MacBeth Witch Scene. Secondly, Ms. Daniel choice of murky, watery imagery reinforced this "kettles full of water" association. Third, the choice of a touch screen as an interface device was appropriate as the viewers could sit on a chair, gaze downwards at the images, and rub their fingers against the screen. In addition, the musical system that was created independently was correspondingly soft in approach, nicely fitting both the imagery and the visual effect.

The last reason for success, which concerns this proposed interactive model, is the mathematical mechanisms that were programmed to map touch to a system response. Since the viewers were engaging the installation with their fingers, it was important to be consistent with this scenario for the interactive visuals. Therefore, we chose the three different visual systems - one for each of the three Easels - to be based on "water ripples", "bending tin", and "rubbing away" visual metaphors. This exposition of process, i.e. water waves reacting to touch, kept the consistency between the manner of interaction – using one's finger – and the visual scenario on which the software was modeled.

With this success, I with to reinforce the importance of consistency between all components of an interactive installation: from the physical structure, to the choice of interface technologies, to the physical references contained within the interaction (e.g. the finger), to the system reconstruction models. In other words, the system behaved in a manner that was consistent with the expectations of the interactive viewer. There was little to "learn" as the interface was relatively intuitive. It was not important that the audience could "reverse engineer" the interactive system, i.e. say "oh, that's a water wave simulation that is based on a grid of springs, that...." All that was important was that there existed a basic fulfillment of their interface expectations. This conclusion is confirmed by the duration of time with which the audience participants would spend experimenting with the *Melody Easels*. Although, as mentioned in Chapter 7, the individual Mind Forest stations were designed with a 3-5 minute experience in mind, it was not uncommon for many viewers to spend up to 10 minutes at one of the Easels. As the interactivity was calm and soothing, an audience member would almost meditatively run their fingers along the touch screen surface.

If there were to be some criticisms of the system, they would be very minor, in my opinion. As each computer for the *Melody Easel* had to run both the image processing and sound generation software, the frame rate - approximately 7 frames per second - was a little bit slower than I would have preferred. Second, the ELO touch screen, although pressure sensitive, required a minimum amount of force before any touch information would be produced, due to the normal "information kiosk" type application that these screens are designed for. This created some situations where the audience member would very lightly rub the screen surface, at which point the ELO would not sense this action, failing to produce any sound or image response. In addition, using strong pressure against the touch screen surface quickly tires the finger, making long and forceful explorations of the *Melody Easels* somewhat exhaustive.

However, both of these complaints are mainly due to the available technology and not to the interactive application. Hopefully, should another round of equipment donations or purchases occur for the *Brain Opera*, these problems could be alleviated.

9.3 Evaluation of the Modified Gesture Walls

Since the modified *Gesture Walls*, as described in Chapter 8, were developed independently of the Brain Opera as part of this thesis project, they have not yet been incorporated into the *Mind Forest* at the time of this writing. It is hoped that during the next leg of *The Brain Opera* world tour, they will become a permanent addition to the production and exposed to a wider test audience.

However, it is possible to comment from my subjective evaluation as an initial starting point. Several key goals were met in this development:

- The frame rate was sped up from 7 frames per second to approximately 22 frames per second (on a 200MHz Pentium Pro), creating a much smoother visual output than was experienced with the Lincoln Center *Gesture Walls*.
- The work took advantage of the local area network that linked each of the computers within the *Mind Forest*, although using only connections between each of the five *Gesture Wall* stations.

- The use of a more robust estimation of the data from the Fish sensors made the system less prone to sporadic errors, smoothing out the system's understanding of the hand motions.
- The visual output was continuous over the entire *Gesture Wall* section, successfully transmitting and receiving particle data as they left the boundaries of each station.
- The total effect of the particle forces was accumulative, allowing for multiple people to build upon the results of the previous participant.

However, there still remain a few issues that were not sufficiently addressed:

- The use of the Fish sensor is still not fully calibrated to each individual user, causing sensing errors based on variations in the viewer's height and weight.
- The body geometry of the viewer is still not properly modeled, as when the participant does not fully reach out with their hand, the sensor data is still biased towards the center of the torso.
- These two above issues still produced sometimes a lack of spatial correlation between the placement of the hand and the region of the visual effect.

Some possible unimplemented solutions to these issues could be to:

- Augment the Fish sensor input with other sensors that could better describe how tall/large a viewer is.
- Provide visual feedback in the video output, such as a "ghostly" graphical hand, that would continuously illustrate where the system believes the viewer's hand to be.
- Use completely relative measurements, thereby losing all spatial information, which would cause other significant problems in the cause-effect relationships in the reconstruction section, i.e. *where* would the graphics be affected?
- Fully calibrate each user by embedding the calibration stage into a "story" that would force each person to place their hands at given locations. The system could then take the Fish measurements at each known point and build a transformation matrix that would better map the raw Fish sensor data to (x,y,z) coordinates.
- Constrain how viewers can stand in front of the *Gesture Wall*. The Sensor Chair is more successful in the Fish measurements because the performer's body is known a priori, relatively fixed, and encourages the performer to reach outwards from the chair.

However, this last set of problem and possible solutions is more related to the input sensing technology than to the general model of interactivity that is proposed. It *does* reinforce the statement that it is important that proper care be taken when implementing the data abstraction stage of the thesis model. False user representation at the beginning of the system pipeline will generally cause large problems throughout the interactive environment.

9.4 Evaluation of Other Interactive Works Based on Model

Two other interactive works of mine, developed outside of the M.I.T. Media Lab and not included in this thesis document, use a similar approach to system design and implementation. An interactive installation, *Winds that Wash*

the Seas (1995), uses a continuous user representations of two tangible interfaces. The first device measures where and how hard a participant is blowing on a computer monitor surface. The second interface measures how water is being stirred in a bathtub. This user representation furnishes control parameters to both an image warp and an alpha-blend algorithm, both calculated in real-time (20 FPS) on a high-end Pentium computer system. This work has been shown at both the MultiMediale 4 (Karlsruhe, Germnay April 1995) and at the ARTEC'97 (Nagoya, Japan June 1997), reaching approximately fifty thousand audience members. Furthermore, it was awarded the Grand Prix award at the ARTEC'97 competition.

A second work, *What Will Remain of These?* (1996), uses a distributed particle system, similar to that which was proposed in Chapter 8. However, the user representation is derived from a real-time optical flow analysis of how people move naturally through an architectural space. The motions of the particle system are coupled with the natural ebb-and-flow of the real people, exploiting the emergent patterns that occur in normal life. Rather than the participant explicitly operating the interactive environment, the computational system is constantly watching these motions, "learning" how people are moving as a collective whole, and then reconstructing these forces as virtual winds that drive the colorful dust particles. This work has been shown at the WRO97 Media Art Biennale (Wroclaw, Poland, April 1997) and also at ARTEC'97. This piece has been very successful, relatively speaking, and has been invited to be shown at the Ars Electronica 97 (Linz, Austria, September 1997), World Wide Video Festival (Amsterdam, Holland, September 1997), and ISEA'97 (Chicago, U.S.A., September 1997).

9.5 Future Directions

Of course, a system model is never complete and is open to further investigation and exploration. Although I am confident that I have proposed a general model of interactivity that is capable of providing for engaging forms of non-narrative, abstract expression, clearly there is still much research and development work left to be performed. As I strive to provide a mathematically well-defined system, the focus of any additional development in this area would be to explore the process of spatial transformations. While it is difficult to provide models of *meaning*, a task more appropriate for philosophers and theorists, I believe that continued development in creating bridges between the communication of meaning and the underlying communications systems will be fruitful. One of the benefits of the computer-based arts is that there is a large formal history of mathematical and technical investigations. If the artist/engineer is able to find corresponding metaphors between the arts and the sciences, the interactive work will thrive as both concepts and the implementations will be easily supported by each other.

9.5.1 Entropy Equilibrium

One new area of investigation I would like to formally explore is the notion of the fulfillment and denial of expectation as an entropy problem stemming from Information Theory as outlined in Shannon and Weaver.[95] Here the entropy of an information signal is a related to the probability that a symbol is expected to occur at a given point in time. As written in Chapter 1, it is important to provide a balance between fulfilling too many of the viewer's expectations immediately versus continually denying their intuitions. The former leads to a boring work while the

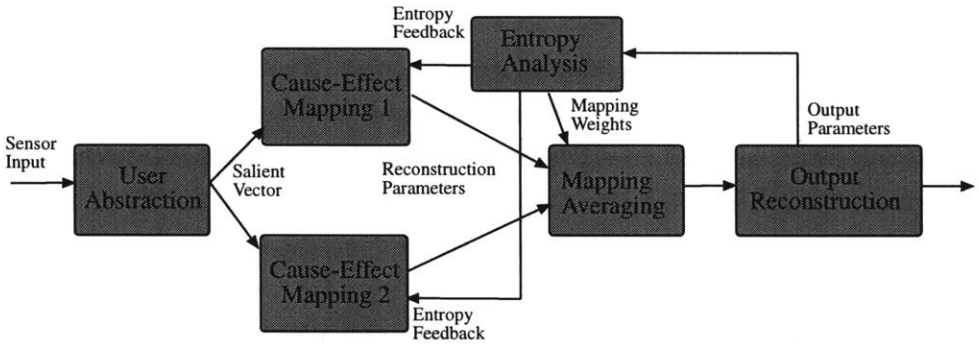

Figure 9-1. Hypothetical system diagram of an entropy-adaptive interactive system. Two "competing" cause-effect mappings are weighted by an entropy analysis in order to balance between different extremes of randomness and predictability.

latter become quickly abstruse and alienating. Therefore, a good work of art should try to balance these two extremes. This goal suggests an idea of "equilibrium" where two contrasting forces find a balance.

Luckily, since we can use the mathematical discourse of communication entropy as outlined by Shannon and Weaver, it may be possible to formally explore the desire to create balances of consonance and dissonance in an interactive environment. In such a case we could consider "competing" interactive systems, S_1 and S_2, each taking the salient user vector, performing a set of independent transformations, and yielding two different sets of output parameter vectors, O_1 and O_2, as outlined in Figure 9-1. These two sets of parameters are passed into a new subsection that would evaluate the results of these output parameters in terms of the entropy values, returning a one-dimensional variable back to the system as feedback. One system would try to maximize this entropy value over time while the other would try to minimize this entropy value. This would hopefully have the corresponding effect of creating two different time-varying experiences, one which is rather harmonious and the other which could appear to be dissonant, in terms of the user's impressions of their expectations being fulfilled or denied. Furthermore, systems S_1 and S_2 could actually be the same system with the identical transformations, but with different behaviors to either minimize or maximize the entropy feedback.

Since these two outputs O_1 and O_2 were formed with this model of interactivity in mind, they are of the same mathematical dimensionality and are output parameter vectors. Therefore it would be trivial to average them together. However, it would be most appropriate to allow different "weightings" to be applied to this averaging, in order to favor one system over the other, i.e. allow either the "harmonious" or the "dissonant" output to have the most influence. Then this averaged output parameter vector could be sent to the reconstruction sub-system of the environment and shown back to the interactive participant.

The hopes in such an investigation would be to see if it may be possible to dynamically adapt to the quality of output based on the artistic concept of consonance and dissonance. If we are successfully able to model this end-user impression, then artists would not have to concern themselves with this issue any longer as it would be invisibly embedded into the system. There are, however, several problems involved in the implementation of such an entropy feedback, as it may prove difficult to form entropy values from the output reconstruction parameters. Therefore, it is

my opinion that such an endeavor would be both a artistic as well as a technological challenge, worthy of further thought.

9.5.2 Further Investigations of Computer Vision Based Interfaces

Although I outlined several salient characteristics that can be derived from computer vision interfaces in Chapter 4, they have not been fully implemented in a working interactive environment. The aforementioned work, *What Will Remain of These?*, uses optical flow analysis from an array of surveillance cameras in order to derive motion estimations that characterize the physical environment. The analysis and reapplication of color and form analysis of images is being explored by Joey Berzowski in order to produce poetic text from images.[96] I would like to extend this area of research to create a richer salient vector of higher mathematical dimensionality that could be used to drive a complex output parameter space. For example, a six dimensional color salient vector could represent the means and variances of each of the three color planes. In addition, we could derive, say, a twenty dimensional salient vector which would be a set of eigenvectors and eigenvalues that roughly describe the form of an interactive viewer. If we also include a salient vector derived from optical flow analysis, perhaps simply the statistical mean of motion vectors, this would form a twenty-eight dimensional user representation. Thus, a very large hyper-space could be formed which contains all of the possible representations of the interactive participant.

The challenge, and it is a difficult one indeed, is to create a set of space transformations between the user representation and the output parametric system. The systems that were presented in this thesis document used a relatively small and manageable number of mathematical degrees of freedom. As the number of degrees of freedom increases, it becomes more difficult to provide for space transformations that are pleasing to the viewer and are mathematically well formalized. One could imagine the difficulties involved in creating a mapping function that projects a twenty-eight dimensional vector into a, say, a three dimensional output parameter space!

9.5.3 Further Investigations of Physical Based Modeling

Two main types of physical-based modeling – the simplified spring and particle systems as described in Chapter 7 – have been explored by this thesis work. It would be interesting to continue to develop real-time simulations of such physical based models and find other applications for their use. Several "real-life" components of the particle system were not implemented due to real-time requirements, such as particle collision detection and having a 3D visualization environment. Furthermore, I would also like to explore the technical and creative applications of implementing additional inter-particle forces, such as attraction and repulsion forces. This could be easily implemented though the assignment of a continuous valued "charge", say between -1.0 and 1.0, to each particle. Particles of similar charge polarity will yield repulsive forces against each other, in proportion to the magnitude of their charge as well as inversely proportional to the distance between them. Conversely, particles of opposite polarity would attract each other. Either the charge values could be assigned by the system automatically or the charges are the result of the user's interactions.

It would be interesting to see how particles "clump" together, balancing all of the forces that are active in the environment: momentum, friction, user forces, and the new inter-particle forces. Unfortunately, although such a

system would be rather straightforward to implement, my initial suspicions are that it would require too much compute power to perform in real-time. This is due to the fact that every particle must compute these inter-particle force calculations against every other particle, giving the system become an "expensive" $\mathbf{O}(N^2)$ compute complexity. However, by making compromises - such as limiting the force calculations to within a smallish neighborhood - it may be feasible to use these extensions in a real-time setting.

9.6 Conclusions

In this thesis project and document, I have presented a general mechanism for the design and implementation of a computer based art form whose intention is to enable the communication of a non-narrative, abstract experience. I consider this work to be purely "computational", as it would not be possible to create such systems outside of the computer. The computer is used to create a set of models - user, thematic, community, and reconstruction – that can be algorithmically simulated in the interactive environment. By choosing not to include the narrative genres of interactive works, it is possible to view the entire system as a set of mapping functions that translate input stimulus into reconstructed responses.

This model was applied to several test cases: six *Interactive Forest* experiences as part of Tod Machover's *The Brain Opera* world-touring show. The interactive installations that were developed sought to emphasize a subtle, abstract quality of "presence", in which the viewer continuously alters the visual material. Since the model uses continuous parametric descriptions of the viewer as well as corresponding mathematical transformations, it was easy to change from one apparent system behavior to another through very minor changes to the transformation parameters. As described in Chapter 7, it was possible for two different environments to be based on exactly the same model except for a single sign change, yielding two strongly different interactive qualities, i.e. from "scatter" to "swarm". Likewise, by dropping a single term in the equations, a "sweep" interactive behavior emerged. Once again small, easy to implement changes to the system yielded a substantial variety in the experience.

Furthermore, such a model has proven to be scalable, as it was possible to gradually increase the complexity of the experience while still developing on top of the same model. This was illustrated in Chapter 8, where the *Gesture Walls* were altered to include a network link between neighboring stations although almost all of the underlying mathematical systems were the same! The time needed to implement the changes between the two versions of the *Gesture Walls* was minimal, since it was necessary to only add functionality on top of that which was already present. It is my strong belief that such a variety of experiences from the same model is unique. Usually, in order for a state-space based work to accommodate such a drastic change of experience, the entire underlying state-space would most likely have to be re-written by the artist.

It is my hope that - through this thesis - I have been able to demonstrate how mathematics and artistic expression can exist side-by-side, where one discipline complements the other. Rather than the artist using the programmer to implement his/her ideas or the scientist restricting the aesthetics of the artist, it is my wish that in the near future the division between programmers and artists will gradually vanish. With this document, I hope to have demonstrated that there is indeed a consistency between these two worlds and that additional research and development within these areas - and especially where they interconnect - are well founded.

References

[1] Dodge, C., *What Will Remain of These?*, interactive installation, Ars Electronica 97 proceedings, Linz, Austria, September, 1997

[2] Klotz, H., "Video Art", *Mediascape Exhibition Catalog*, Guggenheim, New York, 1997

[3] Penny, S., "Consumer Culture and the Technological Imperative: The Artist in Dataspace", Critical Issues in Electronic Media, State University of New York Press, 1995

[4] Laurel, B., Computers as theatre, Addison-Wesley, 1991

[5] As quoted by Octavio Pax, The Castle of Purity, Viking Press, 1978

[6] Mitchell, W., The Reconfigured Eye, MIT Press, 1992

[7] Weinbren, G., *Sonata*, Iterations Exhibition Catalog, The MIT Press 1993

[8] Waliczky, T., as said in WRO'97 symposium lecture, Poland, April 1997

[9] Davenport,G., B. Bradley, "Indexes Are Out, Models Are In", Visions and Views, *IEEE Multimedia*, Fall, vol. 4, no. 3.

[10] http://wwmm.www.media.mit.edu/

[11] Davenport, G., S. Agamanolis, "At the Edge of Dream World: Media Encounters in Architectural Venues", ISEA'97 symposium, Sept. 1997

[12] Scott, J., *Frontiers of Utopia* , interactive installation, MultiMediale 4 Exhibition Catalog, Zentrum für Kunst und Medientechnologie, Karlsruhe, Germany 1995.

[13] Davis, J., "Appearance-Based Motion Recognition of Human Actions", M.S. Thesis for Media Laboratory, June 1996

[14] Habermas, J., "Modernity - An Incomplete Project", The Anti-Aesthetic: essays on postmodern culture, Bay Press, Seattle, 1983

[15] "The Second Modernism", Symposium on Media Art, MultiMediale 4 Exhibition Catalog, Zentrum für Kunst und Medientechnologie, Karlsruhe, Germany 1995.

[16] New World Graphics, computer graphics modeling libraries, http://www.nwginc.com/OFFICPR.HTM

[17] Resnick, M., Turtles, Termites, and Traffic Jams, MIT Pess, 1997

[18] Frances Dyson, "In/Quest of Presence," in Critical Issues in Electronic Media, Simon Penny editor, (Albany, NY: State University of New York, 1995

[19] Waliczky, T., *The Garden*, video animation, WRO'93 Exhibition Catalog, Broslaw, Poland 1993.

[20] Waliczky, T., *The Forest*, interactive CD-ROM work, ArtIntact 2, Zentrum für Kunst und Medientechnologie, Karlsruhe, Germany 1995.

[21] Waliczky, T., *The Way*, video installation, MultiMediale 4 Exhibition Catalog, Zentrum für Kunst und Medientechnologie, Karlsruhe, Germany 1995.

[22] Wheale, N., The Postmodern Arts, Routledge, New York, 1995

[23] Eagleton, T., The Illusions of Postmodernism, Blackwell Publishers, 1996.

[24] Baudrillard, J., "The ecstasy of communication", The Anti-Aesthetic: essays on postmodern culture, Bay Press, Seattle, 1983

[25] Seaman, B., "Re-embodied Intelligence", ISEA97 Conference, Chicago, 1997

[26] Rokeby, D., "Transforming Mirrors", Critical Issues in Electronic Media, State University of New York Press, 1995.

[27] Hoberman, P., Ars Electronica 1996 Symposium Proceedings, Linz, Austria, 1996

[28] Bove, V. M. Jr., ''Multimedia Based on Object Models: Some Whys and Hows," IBM Systems Journal, 35, 1996, pp. 337-348.

[29] Becker, S., V. Michael Bove Jr., "Semiautomatic 3-D Model Extraction From Uncalibrated 2-D Camera Views", *SPIE Symposium on Electronic Imaging: Science & Technology*, San Jose, 1995.

[30] Rokeby, D., *Very Nervous System*, Ars Electronica 1991 Exhibition Catalog, Linz, Austria, 1991

[31] Matsumoto, Y., "Schwerkraft und Gnade", *MultiMediale 4 Exhibition Catalog*, Zentrum für Kunst und Medientechnologie, Karlsruhe, Germany 1995.

[32] Jeremienko, N., *Suicide Box*, videotape, 1995

[33] Moghaddam, B., Alex Pentland, "Face Recognition Using View-Based and Modular Eigenspaces", *SPIE Automatic Systems for Identification & Inspection of Humans,* 1995

[34] Wren, C., Ali Azarbayejani, T. Darrell, A. Pentland, " Pfinder: Real-Time Tracking of the Human Body", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, July 1997, vol 19, no 7, pp. 780-785

[35] Maes, P., T. Darrell, B. Blumberg, A. Pentland, " The ALIVE system: Full-body interaction with animated autonomous agents", Vision and Modeling Tech Report TR#257, MIT Media Lab, 1994.

[36] Sparacino, F., "DirectIVE: Choreographing Media Creatures for Interactive Virtual Environments", M.S. Thesis, MIT Media Laboratory, 1996.

[37] Spasova, M., *Sibyl*, interactive installation, *MultiMediale 4 Exhibition Catalog*, Zentrum für Kunst und Medientechnologie, Karlsruhe, Germany 1995.

[38] Metois, E., "Musical Sound Information: Musical Gesture and Embedding Synthesis", Ph.D. Thesis for MIT Media Laboratory, October 1996.

[39] Machover, T., "Brain Opera Update, January 1996", Internal Document, MIT Media Laboratory, 1996

[40] Oliver, W., "The Singing Tree: A Novel Interactive Musical Experience", S.M. Thesis for MIT Media Laboratory, June 1997.

[41] Yu, J. C., "Computer Generated Music Composition", S.M. Thesis for MIT Electrical Engineering and Computer Science Department, May 1996.

[42] Rowe, R., Interactive Music Systems, Machine Listening and Composing, The MIT Press, Cambridge, 1993.

[43] Paradiso, J., N. Gershenfeld, "Musical Applications of Electric Field Sensing", Computer Music Journal, Summer 1997.

[44] Sermon, P., *Telematic Dreaming*, an interactive video installation, ISEA'94 Exhibition Catalog, Helsinki, 1994.

[45] Shaw, J., *The Legible City*, Mediascape exhibition catalog, Guggenheim Museum, New York, 1996.

[46] Ishii, H., B. Ullmer, "Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms", ACM CHI'97 Conference Proceedings, Atlanta, March 1997

[47] Gorbet, M., M. Orth, "Triangles: A Physical/Digital Construction Kit", DIS 97 Proceedings, Amsterdam, August 1997

[48] Paradiso, J., F. Sparacino, "Optical Tracking for Music and Dance Performance", 4th Conference on Optical 3D Measurement Techniques, ETH, Zurich, September 1997

[49] Marrin, T., "Toward an Understanding of Musical Gesture: Mapping Expressive Intention with the Digital Baton", S.M. Thesis for MIT Media Laboratory, June 1996

[50] Paradiso, J., et al, "The Magic Carpet: Physical Sensing for Immersive Environments", ACM CHI'97 Conference Extended Abstracts Proceedings, Atlanta, 1997

[51] Davenport, G., Friedlander, L. (1995) "Interactive Transformational Environments: Wheel of Life." Contextual Media: Multimedia and Interpretation, Chapter 1, pp. 1-25, MIT Press. Cambridge, MA.

[52] Agamanolis, S., "High-level scripting environments for interactive multimedia systems", S.M. Thesis for MIT Media Laboratory, May 1994.

[53] Baird, F., "Tilting at a Dreamer's Windmills", CaiiA Consciousness Reframed, Wales, July 1997

[54] Agamanolid, S., Westner, A., Bove, V. M., "Reflection of Presence: toward more natural and responsive telecollaboration". SPIE Voice, Video and Data Communications Conference, Dallas, November 1997

[55] Agamanolis, S., "High-level scripting environments for interactive multimedia systems", S.M. Thesis for MIT Media Laboratory, May 1994

[56] Pinhanez, C., K. Mase, A. Bobick, "Interval Scripts: a Design Paradigm for Stroy-Based Interactive Systems", Proc. of ACM CHI'97, Atlanta, March 1997

[57] Seaman, B., "Re-embodied Intelligence", ISEA'97 Conference, Chicago September 1997

[58] Seaman, B., *The World Generator*, DEAF 96, Netherlands 1996.

[59] Gorbet, M., M. Orth, "Triangles: A Physical/Digital Construction Kit", DIS 97 Proceedings, Amsterdam, August 1997

[60] Therrien, C., *Decision,* Estimation, and Classification, John Wiley and Sons, New York, 1989.

[61] Iyengar, G., "Information Theoretic Measure for Encoding Video", M.S. Thesis for Media Laboratory, September 1995

[62] Legrady. G., "Image, Language, and Belief in Synthesis", Critical Issues in Electronic Media, State University of New York Press, 1995

[63] Adelson, E., spoken during lecture at Mitsubishi Electric Research Laboratory, August 1996

[64] Foley, J., A. Van Dam, S. Feiner, J. Hughes, Computer Graphics: Principles and Practice, 2nd Edition, Addison Wesley, 1990

[65] Pratt, W., Digital Image Processing, John Wiley & Sons, 1978

[66] Freeman, B., et al, "Computer Vision for Interactive Computer Graphics", Mitsubishi Electric Research Lab, submitted for publication, July 1997

[67] ibid

[68] Massey, M., "Storytelling with Salient Stills", M.S. Thesis, MIT Media Lab, August 1996

[69] Bergen, J., P. Burt, R. Hingorani, S. Peleg, "Multiple Component Image Motion: Motion Estimation", David Sarnoff Research Center, January 1990

[70] Horn, B., *Robot Vision*, MIT Press, Cambridge 1986

[71] Machover, T., *Media Medium*, Musical Score. Milan/Paris: Ricordi.

[72] Waxman, D., "Digital Theremins: Interactive Musical Experiences for Amateurs Using Electric Field Sensing", MS Thesis, Media Lab, MIT, 1995.

[73] Paradiso, J., Gershenfeld, N., "Musical Applications of Electric Field Sensing", Computer Music Journal, Summer, 1997.

[74] Goldgerg, K., et al., *Telegarden*, interactive installation, SIGGRAPH 95, Los Angeles 1995

[75] Dunne, A., F. Raby, *Fields and Thresholds*, Doors of Perception 2, November 1994. http://www.mediamatic.nl/Doors/Doors2/~DunRab/DunRab-Doors2-E.html

[76] Lippman,A. ,R.G.Kermode, "Media Banks: Entertainment and the Internet", *IBM Systems Journal*, Vol. 35, No. 3&4, 1996

[77] Donath, J., " Inhabiting the Virtual City: The Design of Social Environments for Electronic Communities", Ph.D. Thesis for MIT Media Laboratory, 1997

[78] Fleischmann, M.,W. Strauss, "Liquid Views - Another Story of Narcicuss", ISEA'97, September 1997

[79] Krueger, M., Artificial Reality II, Addison-Wesley, 1991

[80] Dodge, C., *Winds that Wash the Seas*, interactive installation, MultiMediale 4 Exhibition Catalog, Zentrum für Kunst und Medientechnologie, Karlsruhe, Germany 1995

[81] Dodge, C., *What Will Remain of These?*, interactive installation, Ars Electronica 97 Exhibition Catalog, Linz, Austria 1997

[82] as written in *Wired*, Issue 5.05, 1997

[83] Snibbe, S., *Motion Phone*, interactive installation, Ars Electronica 1996 Exhibition Catalog, Linz, Austria, 1996

[84] Foley, J., A. Van Dam, S. Feiner, J. Hughes, *Computer Graphics: Principles and Practice*, 2nd Edition, Addison Wesley, 1990

[85] Mann, S., R. Picard, "Virtual Bellows: constructing high-quality images from video", *Proceedings of the IEEE first international conference on image processing*, Austin, Texas, November 1994

[86] Foley, J., A. Van Dam, S. Feiner, J. Hughes, *Computer Graphics: Principles and Practice*, 2nd Edition, Addison Wesley, 1990

[87] Press, W., W. Vetterling, S. Teukolsky, B. Flannery, *Numerical Recipes in C*, 2nd Edition, Cambridge University Press, 1992

[88] Cook, P., "A Heirarchical System for Controlling Synthesis by Physical Modeling", ICMC, Banff, Canada, Sept. 1995

[89] Glassner, An Introduction to Ray Tracing, Academic Press, London, 1989

[90] Resnick, M., Turtles, Termites, and Traffic Jams: Explorations in Massively Parallel Microworlds, MIT Press, 1994

[91] W. T. Reeves, "Approximate and Probabilistic Algorithms for Shading and Rendering Structured Particle Systems", Computer Graphics, vol. 19, no. 3, pp 313-322, 1985.

[92] Forsythe, W., *Improvisational Technologies*, CD-ROM produced by Zentrum für Kunst und Medientechnologie, 1995

[93] Minsky, M., Society of Mind, Simon and Schuster, New York, 1988

[94] Machover, T., "Brain Opera Update, January 1996", Internal Document, MIT Media Laboratory, 1996

[95] Shannon, C., B. Weaver, The Mathematical Theory of Communication, University of Illinois Press, 1949

[96] Berzowski, J., "Velcro Dreams", CaiiA Consciousness Reframed, Wales, July 1997